The 8th International Meeting
# of the
HP General Systems
Users Group

PROCEEDINGS

Ecole Supérieure de Commerce de Lyon
France
Septembre 1979

QA76.8
H19 H19
1979

ABSTRACTS AND PROCEEDINGS OF THE
HP GENERAL SYSTEMS USERS GROUP
EIGHTH INTERNATIONAL MEETING

September 4 thru September 7, 1979
ECOLE SUPERIEURE DE COMMERCE DE LYON
Lyon, FRANCE

11-14-80 Library Donation

TABLE OF CONTENTS

		Index Number
I	- Programme	Page 1
II	- List of sessions in chronological order	Page 5
III	- Index by Author	Page 8
IV	- Index by Author's Employer	Page 9
V	- Abstracts	Code A
VI	- Proceedings	Code P
VII	- Conference Participants	-
VIII	- Conference Exhibitors	-

N.B. : We apoligise for the one or two sessions with no corresponding abstracts or proceedings.

HPGSUG 8th INTERNATIONAL MEETING, LYON, 4-7 SEPTEMBER 1979

OUTLINE PROGRAMME

TUESDAY: 4 SEPTEMBER

Time	Auditorium	Amphi 0	Amphi 1
09.30-10.30	HPGSUG Business Meeting	-	-
10.30-11.00	BREAK		
11.00-12.30	'Process Management' Chris Moeller HP Cupertino	'Innovation in user designed non-image data base management' David Brown Nice Corporation 'SPL programming techniques for Business Applications' Bruce Toback A.L.S. Corporation	'An integrated system for documentation maintenance program & job selection and memo processing' John Schick Wm. C. Brown & Co. 'Interactive Project Analysis, & Control System' Jeremy Smithers London Business School
12.30-14.00	LUNCH		
14.00-15.30	'Support for 307X Data Capture Terminals on GSD products' Peter Stuart HP Grenoble 'Development and Implementation of a data collection and control system' Barry Tauber Joe Prapuolenis D.S.I. Inc.	'Conversion of Batch Applications to On-Line Operation on the HP 3000' Klaus-Dieter Klien HP Frankfurt 'A French Accounting Package. An exercise in portability from IBM to HP 3000' Claude Novaresio Thierry Videlaine Sligos-CPP	'Overcoming Operational Management Problems with a Scheduler' Clive Oldfield Kim Harris London Business School 'Hybrid Software for the Automated Office' James Taylor
15.30-16.00	BREAK		
16.00-17.30	HP Update: Products and services	-	-
19.00	RECEPTION GIVEN BY THE MAYOR OF LYON		

WEDNESDAY: 5 SEPTEMBER

Time	Auditorium	Amphi 0	Amphi 1
09.00-10.30	'Image - How does it really work' Wayne Benoit Epsilon Data Management 'Adding new capabilities to Image/3000 by Capturing Transactions' Robert Green Robelle Consulting	'A Distributed Structural Design Graphics System' Lawrence Nolan McDonnell Douglas Automation Co. 'GPGS-F, a Portable Device Independent Graphic System' Fredrik Major Ditlef Martens The Ship Research Inst of Norway	'Performance and Architecture Latest HP 3000' Elik Porat HP Cupertino
10.30-11.00	BREAK		
11.00-12.30	'Using Image in a Polydimensional Mode' Jim Russell Computer Services Corporation 'The Data Base Administrator: A Ruthless Juggler of Reality, Algorithms and Data Structures Alfredo Rego Rego Software Pty	'Systems Development' J. G. Oliverio GTE Data Services 'Introduction of PEARL on HP Computer Systems' J. Grossler Technische Universitat Berlin	'Terminal Communications' Ed Turner HP Cupertino 'Performance Measurement for Capacity Planning' Mats Jonsson HP Sweden Robert Abehassera HP France
12.30-13.30	LUNCH		
13.30-14.30	Keynote Speaker Michel KOUTCHOUK Application Development Manager AIR FRANCE 'Logical Construction of programs (M.Warnier) and use in airline company'		
14.30 -18.00	Vendor Show Video tape: 'Memory Manager'; Wendell Henry: HP Cupertino: HPGSUG, Denver. Small rooms will be available for ad hoc groups, 'birds of a feather' sessions, etc.		

THURSDAY: 6 SEPTEMBER

Time	Auditorium	Amphi 0	Amphi 1
09.00-10.30	'Next IT Release' Ray Johnson HP Cupertino 'Support Services' Ilene Birkwood HP Cupertino Bob Lewin HP Geneva	'The Effective Implementation of an Interpretative Operating System' Chris Fugitt HP Boblingen 'HP's Role as a Supplier of Application Software for the HP 250' Heiner Schroder HP Boblingen	'QUIZ - A generalised report writer for the HP 3000' Robert Minns Fred Smith Quasar Systems Ltd 'REX 3000 - A report writer for users and programmers' Lance Carnes Gentry Inc.
10.30-11.00	BREAK		
11.00-12.30	'The changing world of Cobol' Greg Gloss HP Cupertino	'Workshop for HP 250 OEM's' Maurizio Gasperi HP Boblingen	'How to design a data base' Dick Breon Tasco 'Financial Planning on the HP 3000: How the Dollar-Flow language helps users to write their own applications' Jack Damm The Palo Alto Group
12.30-14.00	LUNCH		
14.00-15.30	'ASK/3000 The necessary complement to IMAGE/QUERY' Pierre Senant Cogelog		'A Direct Mail Addresses Application in a Transaction Processing Environment with short Response Time Requirements' Toivo Tagel - DAFA
15.30-16.00	BREAK		
16.00-17.00	HP Senior Management Round Table	-	-

BANQUET AT CHATEAU DE CORCELLES

FRIDAY: 7 SEPTEMBER

Time	Auditorium	Amphi 0	Amphi 1
09.00-10.30	'Data communications with the HP 3000 in Europe' Heinz Jurack HP Boblingen	'Private Volumes & User Logging' Ken Spalding HP Cupertino	'HP 300 - Product Evolution and Direction' Walter Utz HP Cupertino
10.30-11.00	BREAK		
11.00-12.30	Extra commentary by Ed Turner 'Terminal Communications'	'HP Software Development in manufacturing' Rich Zalisk HP Cupertino John Stark HP Bellevue 'Interfacing quantitative techniques to MFG 3000' John Stark HP Bellevue	'Workshop for HP 300 OEM's' Dave Iuppa HP Boblingen
12.30-14.00	LUNCH		
14.00	AU REVOIR OPTIONAL VISIT TO HP GRENOBLE PLANT.		

LIST OF SESSIONS IN CHRONOLOGICAL ORDER

Index Number

Process Management 1
Chris Moeller
Hewlett-Packard - Cupertino

Innovation in user designed non-image data base management 2
David Brown
Nice Corporation

SPL programming techniques for business applications 3
Bruce Toback
A.L.S. Corporation

An integrated system for documentation maintenance program & job selection and memo processing 4
John Schick
Wm. C. Brown & Co.

Interactive Project Analysis & Control System 5
Jeremy Smithers
London Business School

Support for 307X Data Capture Terminals on GSD products 6
Peter Stuart
Hewlett-Packard - Grenoble

Development and Implementation of a data collection and control system 7
Barry Tauber and Joe Prapuolenis
D.S.I. Inc.

Conversion of Batch Applications to On-Line Operation on the HP 3000 8
Klaus-Dieter Klien
Hewlett-Packard - Frankfurt

A french accounting Package. An exercice in portability from IBM to HP 3000 .. 9
Claude Novaresio and Thierry Videlaine
Sligos - CPP

Overcoming Operational Management Problems with a Scheduler 10
Clive Oldfield and Kim Harris
London Business School

Hybrid Software for the Automated Office 11
James Taylor
TAYLOR Consultant

Image - How does it really work 12
Wayne Benoit
Epsilon Data Management

Adding new capabilities to Image/3000 by Capturing Transactions 13
Robert Green
Robelle Consulting

Index Number

A distributed structural Design Graphics System.................................. 14
Lawrence Nolan
Mac Donnell Douglas Automation Co.

GPGS-F, a Portable Device Independent Graphic System........................ 15
Fredrik Major and Ditlef Martens
The Ship Research Inst. of Norway

Performence and Architecture latest HP 3000...................................... 16
Elik Porat
Hewlett Packard - Cupertino

Using Image in a Polydimensional Mode.. 17
Jim Russell
Computer Services Corporation

The Data Base Administrator : A Ruthless Juggler of Reality,
Algorithms and Data Structures.. 18
Alfredo Rego
Rego Software Pty

Systems Development.. 19
J. G. Oliverio
GTE Data Services

Introduction of PEARL on HP Computer Systems................................. 20
J. Grossler
Technische Universitat - Berlin

Terminal Communications.. 21
Ed Turner
Hewlett Packard - Cupertino

Performance Measurement for Capacity Planning................................ 22
Mats Jonsson
Hewlett Packard - Sweden
Robert Abehassera
Hewlett Packard - France

Next IT Release.. 23
Ray Johnson
Hewlett Packard - Cupertino

Support Services... 24
Ilene Birkwood
Hewlett Packard - Cupertino
Bob Lewin
Hewlett Packard - Geneva

The Effective Implementation of an Interpretative Operating System......... 25
Chris Fugitt
Hewlett Packard - Boblingen

HP's Role as Supplier of Application Software for the HP 250............... 26
Heiner Schroder
Hewlett Packard - Boblingen

Index Number

QUIZ - A generalised report writer for the HP 3000........................... 27
Robert Minns and Fred Smith
Quasar Systems LTD

REX 3000 - A Report Writer for Users and Programmers.......................... 28
Lance Carnes
Gentry Inc.

The changing World of COBOL.. 29
Greg Gloss
Hewlett Packard - Cupertino

Workshop for HP 250 OEM's... 30
Maurizio Gasperi
Hewlett Packard - Boblingen

How to design a data base... 31
Dick Breon
Tasco

Financial Planning on the HP 3000 : How the Dollar-Flow language helps users to write their own applications.................................. 32
Jack Damm
The Palo Alto Group

ASK/3000 The necessary complement to IMAGE/QUERY.............................. 33
Yan Cordelle and Pierre Senant
Cogelog

A direct Mail Addresses Application in Transaction Processing Environment with short Response Time Requirements............................ 34
Toivo Tagel
DAFA

Data Communications with the HP 3000 in Europe............................... 35
Heinz Jurack
Hewlett Packard - Boblingen

Private Volumes & User Logging.. 36
Ken Spalding
Hewlett packard - Cupertino

HP 300 - Product Evolution and Direction...................................... 37
Walter Utz
Hewlett Packard - Cupertino

HP Software Development in Manufacturing...................................... 38
Rich Zalisk
Hewlett Packard - Cupertino
John Stark
Hewlett Packard - Bellevue

Interfacing Quantitative Techniques to MFG 3000............................... 39
John Stark
Hewlett Packard - Bellevue

Workshop for HP 300 OEM's... 40
Dave Iuppa
Hewlett Packard Boblingen

Memory Management - A videotape by Wendell Henry - Extra Commentary........... 41
Chris Moeller
Hewlett Packard - Cupertino

INDEX BY AUTHOR

Index Number

Abehassera Robert 22
Benoit Wayne 12
Birkwood Ilene 24
Breon Dick 31
Brown David 2
Carnes Lance 28
Cordelle Yann 33
Damm Jack 32
Fugitt Chris 25
Gasperi Maurizio 30
Gloss Greg 29
Green Robert 13
Grossler J. 20
Harris Kim 10
Iuppa Dave 40
Johnson Ray 23
Jonsson Mats 22
Jurack Heinz 35
Klein Klaus-Dieter 8
Lewin Bob 24
Major Fredrik 15
Martens Ditlef 15
Minns Robert 27
Moeller Chris .... 1 and 41
Nolan Lawrence 14
Novaresio Claude 9
Oldfield Clive 10
Oliverio J.G. 19
Porat Elik 16
Prapuolenis Joe 7
Rego Alfredo 18
Russell Jim 17
Schick John 4
Schroder Heiner 26
Senant Pierre 33
Smith Fred 27
Smithers Jeremy 5
Spalding Ken 36
Stark John .... 38 and 39
Stuart Peter 6
Tagel Toivo 34
Tauber Barry 7
Taylor James 11
Toback Bruce 3
Turner Ed 21
Utz Walter 37
Videlaine Thierry 9
Zalisk Rich 38

INDEX BY AUTHOR'S EMPLOYER

	Index Number
A.L.S. Corporation - Toback Bruce	3
COGELOG - Cordelle Yann	33
COGELOG - Senant Pierre	33
Computer Services Corporation - Russell Jim	17
DAFA - Tagel Toivo	34
D.S.I. Inc. - Prapuolenis Joe	7
D.S.I. Inc. - Tauber Barry	7
Epsilon Data Management - Benoit Wayne	12
GENTRY Inc. - Carnes Lance	28
G.T.E. Data Services - Oliverio J.G	19
Hewlett Packard BELLEVUE - Stark John	38 and 39
Hewlett Packard BOBLINGEN - Fugitt Chris	25
Hewlett Packard BOBLINGEN - Gasperi Maurizio	30
Hewlett Packard BOBLINGEN - Iuppa Dave	40
Hewlett Packard BOBLINGEN - Jurack Heinz	35
Hewlett Packard BOBLINGEN - Schroder Heiner	26
Hewlett Packard CUPERTINO - Birkwood Ilene	24
Hewlett Packard CUPERTINO - Gloss Greg	29
Hewlett Packard CUPERTINO - Johnson Ray	23
Hewlett Packard CUPERTINO - Moeller Chris	1 and 41
Hewlett Packard CUPERTINO - Porat Elik	16
Hewlett Packard CUPERTINO - Spalding Ken	36
Hewlett Packard CUPERTINO - Utz Walter	37
Hewlett Packard CUPERTINO - Zalisk Rich	38
Hewlett Packard FRANCE - Abehassera Robert	22
Hewlett Packard FRANKFURT - Klein Klaus-Dieter	8
Hewlett Packard GENEVA - Lewin Bob	24
Hewlett Packard GRENOBLE - Stuart Peter	6
Hewlett Packard SWEDEN - Jonsson Mats	22
London Business School - Harris Kim	10
London Business School - Oldfields Clive	10
London Business School - Smithers Jeremy	5
Mc Donnell Douglas Automation Co. - Nolan Lawrence	14
Nice Corporation - Brown David	2
Palo Alto Group (the) - Damm Jack	32
Quasar Systems LTD - Minns Robert	27
Quasar Systems LTD - Smith Fred	27
Rego Software PTY - Rego Alfredo	18
Robelle Consulting - Green Robert	13
Ship Research Institut of Norway (the) - Major Fredrik	15
Ship Research Institut of Norway (the) - Martens Ditlef	15
SLIGOS - CPP - Novaresio Claude	9
SLIGOS - CPP - Videlaine Thierry	9
TASCO - Breon Dick	31
TAYLOR Consultant - Taylor James	11
Technische Universitat BERLIN - Grossler J.	20
Wm. C. BROWN and Co - Schick John	4

ABSTRACTS

Presentation Abstract

Presentation Title: OPTIMIZE YOUR BUSINESS DATA PROCESSING USING SPL

Author(s): BRUCE TOBACK

Title(s): SPL Programming Techniques for Business Applications

Address:
A. L. S. Corporation
1400 N. Baxter St.
Anaheim, CA 92806
USA

Phone 714 956-9200

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

In many HP 3000 installations, HP's Systems Programming Language, or SPL, is used very sparingly if at all. Use of the langua-e has been restricted in large part to creating interfaces to MPE intrinsics from COBOL and BASIC. This paper focuses on more general application of SPL to business data processing. The first portion of the paper emphasizes the very real management concerns of using a sophisticated, machine-dependent language in a business environment. Maintainability, portability, and compatibility of SPL modules are some of the issues addressed.

The second portion deals with specific applications for SPL, and discusses the range of situations for which SPL will provide significant benefits over a traditional business language, including space -and time- critical requirements and complex data manipulation tasks.

Finally, the author discusses SPL coding techniques, with special attention given to solutions to EDP problems. The paper provides the reader with a "boilerplate" approach to SPL, similar to the COBOL COPYLIB facility, as well as a series of convenient cookbook SPL solutions to a variety of business data processing problems.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: AN INTEGRATED SYSTEM FOR DOCUMENTATION MAINTENANCE, PROGRAM AND JOB SELECTION AND MEMO PROCESSING

Author(s): JOHN A. SCHICK

Title(s): Senior Programmer

Address:
Wm. C. BROWN Co., Publishers
2460 Kerper Blvd
Dubuque, IA 52001

Phone 319 - 588-1451

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

A system has been written at Wm. C. Brown Co. to meet the following diverse needs :

1. An on-line system to maintain and print program documentation including file, subprogram and copylib usage information and to install programs into the the several production accounts.
2. A program to ease the program selection process by allowing programs to be accessed by mnemonic name or by choosing from a list of program descriptions organized by system.
3. A job streaming utility that eliminates the need for hard-coded account and user names and passwords in job files and allows for the entry of variable data into job streams.
4. A system for the sending, receiving and indexing of memos throughout the company via the computer.

The system is written in SPL employing IMAGE databases and both VIEW and character-mode entry programs. The job streaming utility uses privileged-mode code to access the system directory.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: INTERACTIVE PROJECT ANALYSIS AND CONTROL SYSTEM

Author(s): Jeremy SMITHERS

Title(s): Senior Research Officer

Address: London Business School, Sussex Place, Regent's Park,
LONDON NW1 4SA

Phone (01) 262-5050

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

IPACS is an on-line package for the planning and control of a project. It is written in FORTRAN and designed to operate in a mini-computer time-shared environment. It is based on the technique of network analysis and its major features are ease of use and flexibility of operation. It has sophisticated project control features allowing time recording against individual resources on an activity down to the nearest manhour.

It is based on a modular command structure where each command is a particular function which prompts the user for the required input items. There are two levels of prompts expert and novice. The report generator has an extensive set of facilities for controlling the layout and content of reports. The subnetworking facility allows the project to be subdivided into manageable work units and then a skeleton network may be used for overall control. It also allows standard subnetworks to be loaded from a library. IPACS effectively maintains a database for a project with a record of the original plan (budget) and details of actual progress. Progress may be monitored against the original or a revised budget. It has been designed in an open-ended fashion to allow new facilities to be added easily.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: SUPPORT FOR 307x DATA CAPTURE TERMINALS ON GSD PRODUCTS

Author(s): Session Leader : PETER STUART

Title(s):

Address: HP GRENOBLE

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

This session will include :

- Overview of 3075/76/77 Terminals
- Description of an existing application (HPG Product Tracking)
- Current supported capability on 3000 PT to PT and multipoint
- Discussion on General purpose Data Capture versus Application specific Data Capture.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: DEVELOPMENT AND IMPLEMENTATION OF A DATA COLLECTION AND CONTROL SYSTEM

A SOFTWARE VENDORS VIEW

Author(s): BARRY V. TAUBER / Project Memory

S. JOE PRAPUOLENIS / Implementation Manager

Title(s):

Address:
DSI, Inc.
180 N. MICHIGAN (500)
CHICAGO IL 60601

Phone 312 - 372 - 4222

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

The development and implementation of a shop floor system for a manufacturing concern is presented from the viewpoint of a software house with first time exposure to the HP 3000. Mapping of specifications to the 3000, use of developmental aids, data base design, and testing are presented for this installation. Major success as well as obstacles are highlighted.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: CONVERSION OF BATCH APPLICATIONS TO ON LINE OPERATION ON THE HP 3000

Author(s): Session Leader : KLAUS-DIETER KLEIN

Title(s):

Address: HP FRANKFURT

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The session will begin by defining "Batch" & "On Lin" and examining the fundamental differences between them and the effects of the direct man-machine interaction involved.

The speaker continues, by considering the hardware and system software requirements for successful implementation of On-Line programs and examines the changes needed to the overall organisation.

The question of why one should transfer Batch to On Line is considered and a discussion initiated on the available software tools and how to use them to save time in programming.

The effect on total system performance and response time is also discussed.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: A FRENCH ACCOUNTING PACKAGE
A EXERCISE IN PORTABILITY FROM IBM TO HP 3000

Author(s): a) CLAUDE NOVARESIO
b) THIERRY VIDELAINE

Title(s): a) Product Manager SLIGOS-CPP
b) Former EDP Manager PLASTIC OMNIUM 92 LEVALLOIS FRANCE

Address: SLIGOS-CPP
91, Rue Jean Jaurès
92807 PUTEAUX

Phone 776.42.42

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

See attached notes...

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: A FRENCH ACCOUNTING PACKAGE
AN EXERCISE IN PORTABILITY FROM IBM TO HP 3000

By Claude NOVARESIO
& Thierry VIDELAINE

A - THE PACKAGE

SLIGOS set up GAEL 3 on the bases of knowledge gained from the experience of using a product already existant, that is a general accounting program used in bureau service for several years.

GAEL 3 has the following aims :

- To enable an easy adaptation to the management methods of each company thanks to a highly developed personalization. Input documents, data composition, chart of accounts are customized to the needs of each company.
- To facilitate a rational processing of the general, analytical and budgetary accounting.
- To be able to handle several independant accounting procedures simultaneously.
- To establish an accounting data base containing the data required for specific additionnal applications.
- To be of very easy use.

B - PORTABILITY

The package was successfully running on IBM computers. After chosen by the PLASTIC OMNIUM Company, which had just moved from a IBM System 3 Model 15 to an HP 3000.

The paper describes the procedure used to transfer GAEL 3 from a computer to another, using standard HP Software, and emphasizes the problems encountered as well as the way on which they were solved.

Presentation Abstract

Presentation Title: Overcoming Operational Management Problems with a Scheduler.

Author(s):
Mr. C. H. Oldfield
and
Mr. K. Harris

Title(s):
Computer Manager
and
Operations Supervisor

Address:
The London Business School,
Sussex Place,
Regents Park,
London N.W.1 4SA
England.

Phone
01-262-5050

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

It maybe that some new or existing H.P. 3000 based operations managers are faced with similar problems initially encountered at the London Business School, in attempting to control and allocate limited resources (CPU time, Non-Shareable peripherals etc.) to a wide variety of user needs.

The theme of the presentation will be to give a brief outline on the salient points that convinced us that we required to introduce a scheduling system, to be used as a tool in solving these problems; followed by the design philosphy adopted for the scheduler; continuing through to user reaction on installation; further benefits seen and gained since introduction; examples of scheduler operational details from both user and operator standpoint; and finally proposed future enhancements.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: IMAGE - HOW DOES IT REALLY WORK ?

Author(s): WAYNE BENOIT

Title(s): MGR. Prod. Development

Address: EPSILON DATA MANAGEMENT
24 NEW ENGLAND EXEC. PARK
BURLINGTON Mass. 01803

Phone (617) - 273 - 0250 (306)

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

A presentation of what the root file looks like and its relationship to other image data sets.

Further discussion of utilities and tips on building and maintaining data bases, as well as modifying in place.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: ADD NEW CAPABILITIES TO IMAGE / 3000
BY CAPTURING TRANSACTIONS

Author(s): ROBERT M. GREEN

Title(s): President
ROBELLE CONSULTING Ltd

Address: = 130 - 5421 10th Ave.
DELTA, B. C. V4M 3T9
CANADA

Phone (604) 943 - 8021

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

This paper is addressed to the manager or designer of IMAGE / 3000 on-line applications. I raise questions regarding throughput, integrity, response time, design ease, maintainability and failure recovery.

I suggest the use of an interface between the application programs and the database system to answer some of these problems. One such interface that I have developed, is called "SNAPSHOT" and is currently in use at a test site.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: A DISTRIBUTED STRUCTURAL DESIGN GRAPHICS SYSTEM

Author(s): MR LAWRENCE E. NOLAN

Title(s): SUPERVISOR GRAPHICS PRODUCTS MCAUTO

Address: DSI, INC.
180 N. MICHIGAN AVE. SUITE 2200
CHICAGO, ILLINOIS
60601, USA

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

THE FASTDRAW/3 DISTRIBUTED GRAPHICS SYSTEM INCLUDES COMMANDS TO GENERATE POINTS AND ELEMENTS IN LOCAL COORDINATE SYSTEMS, PERFORM SIMPLE CONSTRUCTION OF ARCS ,CIRCLES AND CREATE ARBITRARY SPACE CURVES AND SURFACES. FASTDRAW/3 CAN AUTOMATICALLY CREATE A MESH OF ELEMENTS ON A DEFINED SURFACE AFTER CONTROL POINTS ALONG THE BOUNDARIES HAVE BEEN SELECTED. ALSO, ELEMENTS CAN BE DUPLICATED BY ANY COMBINATION OF TRANSLATION, ROTATION, AND SCALING IN TWO OR THREE DIMENSIONS. FASTDRAW/3 SUPPORTS USER DEFINED ELEMENTS THAT ALLOW A DESIGN TO BE ACCOMPLISHED IN PARTS AND THE PARTS THEN COMBINED TO BUILD THE FINAL GEOMETRIC MODEL.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: GPGS-F, a portable device inedependant graphic system.

Author(s): Fredrik Major
Ditlef Martens

Title(s):

Address: The Ship Research Institute of Norway
P.O.Box 4125, Valentinlyst
N-7001 TRONDHEIM
Norway

Phone 075-95500

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

The Ship Research Institute of Norway (NSFI) has implemented GPGS-F on the HP 3000 series computer and is responsible for HP versions of the system. GPGS-F is a system near to the proposed ISO-standard for graphic systems and is choosen as standard graphic system by the Norwegian Co-operation in Computer Graphics (NORSIGD). This organization is responsible for the basic development of GPGS-F which is carried out on a Univac 1100 series computer. The system is implemented on many different computers and has device drivers for most of the common graphic devices. The paper will present the features of GPGS-F and describe the HP-3000 implementation. If possible GPGS-F may also be made available on the local HP 3000 during the meeting.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: PERFORMANCE AND ARCHITECTURE LATEST 3000

Author(s): ELIK PORAT

Title(s):

Address: HP GSD-R&D Cupertino

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

See attached notes

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: PERFORMANCE AND ARCHITECTURE LATEST 3000 By Elik PORAT

This workshop is intended to explore and explain the 3000 System family architecture as well as future architecture trends. The performance implications of the various system building blocks (i.e. : CPU, Memory, I/O, Diagnostics) will be discussed.

As a set of interacting primary building blocks which are cohesively interwoven, the 3000 provides optimum system performance. The 3000 system family -with its MPE based operating systems- incorporates a design philosophy stressing modularity which provides for growth potential. System growth can be realized in capabilities, computing power, transaction processing, and distributed networks.

Architectural changes for system enhancement are made transparent to the user via new approaches to microcode. Remote system access and built in diagnostics capabilities are the new features of the 3000 systems family which enhance system uptime and personnel productivity.

LSI technology advances are increasingly playing a major role in architectural design considerations. As a result, several "TRADITIONAL" system building will change shape or disappear altogether as we know them, and much functional integration will take place.

Microcoded macro functions should unburden system resources from time consuming house keeping duties. Smart system peripherals with intellignet controllers, communicating with each other, will affect the structure of both the operating system and the system hardware. Major system functional blocks will be decomposed into functional operations.

Presentation Abstract

Presentation Title: USING IMAGE IN A POLYDIMENSIONAL MODE

Author(s): JIM RUSSELL

Title(s): President, Computer Services Corporation

Address: Computer Services Corporation
75 Manhattan Drive, Suite 107
Boulder, COLORADO, USA 80303

Phone (303) 499-3713

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

See Attached

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: USING IMAGE IN A POLYDIMENSIONAL MODE

By Jim RUSSELL

1 - The scarcest resource in an HP 3000 system is disc 10 - approximately 30 per second equals max.

2 - IMAGE is a limited data base management system providing only a two-cimensional seeking selection technique for programmers-data set and data item.

3 - Because of this limitation, it is frequently necessary to do long serial reads through the data base in order to find the exact entry needed.

4 - Long serial reads of the data base degrade performance for the user as well other users on the system that must queue up for the disc 10.

5 - Therefore, if a data base design can be created which logically expands the capacity of IMAGE from a two-dimensional to a multi or polydimensional capability it would greatly improve performance for the user and all other users on the system by dramatically reducing disc 10s - as much as 98 %.

6 - I have developed a technique in data base design which uses a "combination-item" approach to achieve this result.

7 - Simply stated, one or more "combination-items" are created in the data set so that multiple data items can be searched for by searching directly to the "combination-item" that contains all the intelligence required. This "combination-item is an automatic or manual master and allows for one disc seek in order to find the correct record.

Presentation Abstract

Presentation Title: "THE DATA BASE ADMINISTRATOR : A RUTHLESS JUGGLER OF REALITY, ALGORITHMS AND DATA STRUCTURES"

Author(s): F. ALFREDO REGO

Title(s): Professor Universidad FRANCISCO MARROQUIN
Chief Executive Officer REGO SOFTWARE Pty

Address: Calle del Arco 24
ANTIGUA
GUATEMALA, Central America

Phone (502-2) 32-4336, TWX 4192 TELTRO GU

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)
Abstract: (No more than 200 words)

See attached notes.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: THE DATA BASE ADMINISTRATOR By Alfredo REGO

The data base administrator (DBA) must keep in perfect balance (emotionally and otherwise) while flawlessly maintaining three dynamically-changing computer-model "eggs" whirling through the air at all times :

1 - The reality that the computer model is supposed to reflect,

2 - The algorithms (programs) that manipulate the computer model to produce results,

3 - The data structures that support the computer model.

I would like to conduct this session as an open forum with lots of questions and cross-firing aimed at the DBA's juggling task and its awesome implications.

Therefore, I will present in this paper only some of my opinions on certain topics :

A - The painful choices involved in the selection and maintenance of the appropriate kinds, shapes, weights, and sizes of the computer model "eggs",

B - The smashing catastrophes hatched when the computer-model ''eggs" happen to be unbalanced,

C - The performance-time adjustments that the DBA might wish to make to get the the act together should one of the computer-model "eggs" get out of hand, break, hatch or whatever,

D - The help (or lack thereof) that the DBA might run into. In particular :

- The tools available (if any) to manipulate reality,
- The tools available (if any) to manipulate algorithms,
- The tools available to manipulate data structures. And, specifically for IMAGE / 3000 data bases : DBUNLOAD / DBLOAD, DBREBILD, DBCHANGE, and ADAGER / 3000 (The ADAPTER / MANAGER for IMAGE / 300 data bases).

Presentation Abstract

Presentation Title: SYSTEMS DEVELOPMENT
(Title not determined at this time, will be identified for final paper).

Author(s): L. D. SALYER
J. G. OLIVERIO

Title(s): Systems Manager
Systems Supervisor

Address: G T E Data Services
P. O. Box 1548
TAMPA, FLORIDA 33601

Phone (813) 224-3408
(813) 224-3404

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

See attached letter to John EATON.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: SYSTEMS DEVELOPMENT

By L. D. SALYER
& J. G. OLIVERIO

I. Background
A. Define GTE Data Services (GTEDS) charter with respect to the GTE Telephone Companies.
B. Brief explanation of how GTEDS provides service to the telephone companies.
1) Define Classes of Data Centers
2) Define Telephone Systems Group

II. The Application
A. Explain the function being mechanized.
B. Brief explanation of the Trunk Administration Project.
1) Trunk Administration
2) TARS

III. Establishing the Requirements
A. The Requirements Team
1) Membership
2) Charter
3) Requirements Document (RQD)
B. The GTEDS proposal to meet the requirements (CNAS)
1) Modularity of development
2) Implementation approach
a) Large mainframe vs. mini-computer
b) Considerations for alternatives
3) Use of IDEA in determining feasibility
C. Future Directions

IV. System Development - First Release (CNAS R1V1)
A. Detail System Design
1) Tops down design methods used
a) HIPO
b) Team concept
c) Data entry philosophy

      2) Data Base Design
         a) Single vs. multiple data bases - considerations
         b) Establishment of logical relationships via applications programming
      3) Hardware requirements
   B. Obtaining user approval
      1) System Specification Report
      2) Use of HP 2645 terminal with cassettes to demonstrate system
   C. System Engineering
      1) Continuation of team concepts in program development
      2) Program design
         a) Mainline concepts
         b) On-line/batch processes
      3) Technical Considerations
         a) HP 3000 System (MPE) limitations
         b) IMAGE/Query constraints
         c) COBOL experiences
         d) Use of SPL
      4) The final product
         a) Lines of code
         b) Schema and capacities
         c) Testing experiences
V. Implementation
   A. User training
   B. Documentation
      1) GTE Practices (GTEPS)
      2) System Design Manual
   C. Conversion experience
   D. Pilot results
      1) User approval
      2) Benefits identified

VI. The Next Step

A. GTE System wide implementation plans

B. Development of the next release

C. Plans for support using HP3000 capabilities

Presentation Abstract

Presentation Title: PERFORMANCE MEASUREMENT FOR CAPACITY PLANNING

Author(s): Session Leader : MATS JONSSON, Stockholm
Assisted by : ROBERT ABEHASSERA, Paris

Title(s):

Address: HP Systems Engineering Organisation

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

This workshop session will cover an overview of existing performance measurement and optimisation tools, and the concept of capacity planning.

Experience gained with customers in Sweden, France and Germany will be discussed.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: THE EFFECTIVE IMPLEMENTATION OF AN INTERPRETATIVE OPERATING SYSTEM

Author(s): Session Leader : CHRIS FUGITT

Title(s):

Address: HP Böblingen

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The speaker will describe the fundamental considerations involved in the design of the HP 250 Operating system and discuss the advantages of the approach taken for a system intended to be primarily an entry level system for first time users in small and medium sized organisations.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: HP'S ROLE AS A SUPPLIER OF APPLICATION SOLTWARE FOR THE HP 250

Author(s): Session Leader : HEINER SCHRODER

Title(s):

Address: HP BOBLINGEN

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The session will begin by reviewing the design considerations involved with the HP 250 and describing the market place for the system.

An overview of the application requirements of these users will lead into a description of the application packages that have been developed by HP for Financial Accounting and Manufacturing.

A discussion will then take place on the usefulness and effectiveness of these packages to the 3rd Party Software Suppliers and to the End Users.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

HP's Role as a Supplier of Application Software for the HP 250

The HP 250 is the entry level system in Hewlett Packard's range of small business computers. It is designed to meet the requirements of the first time user either in a small business or in a department of a large firm.

The marketing strategy for selling the HP 250 to small business is to utilize the expertise of third party software suppliers. To ensure third parties can minimize their initial investment HP 250 application software provides the backbone of all accounting and manufacturing requirements.

The general features of all HP 250 software packages are

(1) Extensive use of System Tools (IMAGE/250, SORT/250, FORMS/250, REPORT WRITER/250) allow easy customization to special user requirements. They also facilitate enhancements to the software because of the relatively little additional programming effort.

(2) All software packages are highly parameter driven. This allows the OEM to tailor the software to specific customer needs without modifying the code.
These parameters can be values within a company which are changing very seldom like formatting a balance sheet or terms of payment.
But it may be also required to customize things very often e.g. have open invoice report either sorted by invoice number or customer number or print only a subset of the whole report.

(3) The user interface is very friendly. The dynamically labelled softkeys guide the user through the program without need of a user manual. The layout of the screen is the same in all programs (Data input area, command input, error message line etc.) so that the user is very quickly able to operate the system.

(4) The software packages are well structured for enhancements.
The use of structured programming, standard subroutines as well as normed programming makes it very easy to understand and change existing programs. Well defined interfaces provide entry points for additional software modules.

(5) The application software is well documented from a user standpoint as well as from a programmer standpoint.

For more information like price, description and availability of the software packages please contact your local HP Sales office because they may be different in the various countries.

Presentation Abstract

Presentation Title: QUIZ - a generalized report writer for the HP3000

Author(s): Robert Minns
Fred H. Smith

Title(s):

Address: Quasar Systems Ltd.,
99 Bank Street
Suite 641
Ottawa,Ontario (CANADA)
KlP 6B9

Phone (613) 237-1440

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

QUIZ is a generalized reporting and inquiry program for the HP3000. Simple ad-hoc reports can be produced by an end-user through entry of minimal specifications. Complex reports may be produced by entry of detailed format specifications. QUIZ operates on any combination of IMAGE/3000 databases, KSAM/3000 files and conventional MPE files.

This paper reviews our experiences in the design and implementation of QUIZ. Particular emphasis is placed upon the difficulties of designing a facility which is simple enough to be used by non-EDP personnel to produce simple reports, but which is also capable of the sophisticated formatting and control required of production reports. User experience in a number of installations is also described.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: GREAT/3000 - A REPORT WRITER FOR USERS AND PROGRAMMERS

Author(s): Lance Carnes

Title(s): Project Manager, GENTRY, INC.

Address: 609 Kearney St.
El Cerrito, CA 94530
USA

Phone (415) 527-4451

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ **(minutes)**

Abstract: (No more than 200 words)

This paper presents a language, GREAT/3000, which was developed to meet the need of non-programmers and programmers to write reports. It discusses the language design decisions which had to be made to meet the requirements of the two user classes. It addresses also the issue of whether the resulting language is translatable by a low-cost, easily maintained compiler. An attempt is made to assess the utility of the resulting language, and to notice whether the compromises in the design either helped or hindered the original purpose of the language. In conclusion, the experiences of the first users of the translator and language will be related, showing that in fact the goal was achieved.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: THE CHANGING WORLD OF COBOL

Author(s): GREG GLOSS

Title(s):

Address: HP GSD Cupertino

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The specifications of the COBOL language, both the ANSI standard and the HP implementation, are continuously evolving.

This paper covers some of the new features in ANSI COBOL - 74 along with how HP COBOL users will be able to convert.

The directions that the ANSI COBOL committee is taking with the next version of the COBOL standard are also discussed.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: WORKSHOP FOR HP 250 OEM'S

Author(s): Session Leader : MAURIZIO GASPERI

Title(s):

Address: HP Böblingen

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The session will provide an opportunity to existing and interested OEM'S to discuss the HP 250's role as HP's entry level Business System.

The topics to be covered are expected to include applications already programmed, the needs of first time users and whether these can best be met by system software or application software.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: FINANCIAL PLANNING ON THE HP 3000 : HOW THE DOLLAR-FLOW LANGUAGE ENABLES USERS TO WRITE THEIR OWN APPLICATIONS.

Author(s): JACK DAMM, The PALO ALTO Group

Title(s): Principal / Owner

Address: The PALO ALTO GROUP

790 Lucerne Drive
Sunnyvale, California 94086
U. S. A.

Phone (408) 735-8490

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

See attached.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

THE PALO ALTO GROUP
790 Lucerne Drive • Sunnyvale, CA 94086 • (408) 735-8490

FINANCIAL PLANNING ON THE HP3000

How the Dollar-Flow language enables end-users to write their own applications.

This presentation deals with two topics: 1) The human engineering involved in the implementation of Dollar-Flow: 2) The application of Dollar-Flow to financial planning.

Financial planning, by its very nature involves change. This fact, combined with the need for immediate results, makes it imperative that the programmer/analyst be eliminated as middleman. Instead, users must be able to use the computer directly to set up their own applications. Techniques employed in Dollar-Flow to make this possible: generalized file structure and the elimination of the distinction between files, programs, and/or reports. Effective language design. Understandable prompts, editing responses, coherent error messages. Training and support. Users manuals and system documentation. The psychology of the terminal user.

Profit planning and control with Dollar-Flow: the need for planning and control. Overall company planning: budgets, sales forecasts, profit/loss statements, cash flow and balance sheet projections. Low-level planning: pricing decisions, capital budgeting, new product decisions. Performance reporting for control: budget variance reporting, product sales variances, comparison of forecast versus actual performance including the use of graphics.

Presentation Abstract

Presentation Title: ASK/3000, a real Image data-base interrogation and reporting language.

Author(s): PIERRE SENANT &YANN CORDELLE
COGELOG

Title(s):

Address: 1 résidence des Quinconces
91190 GIF SUR YVETTE
FRANCE

Phone (1) 907.03.86
907.70.79

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☐ (minutes)

Abstract: (No more than 200 words)

Enclosed

Please return this form to: ~~John Eaton~~
~~London Business School~~
~~Sussex Place~~
~~Regent's Park~~
~~LONDON NW1 4SA England~~

Mme CASSIER Dominique
Ecole Supérieure de Commerce de Lyon
23, route de Dardilly
69130 ECULLY

by Friday 20 April

ABSTRACT : ASK/3000

ASK/3000 is an interrogation and reporting language developped by COGELOG for a more complete utilisation of Image data-base possibilities.

ASK/3000 philosophy is the same as Query, but almost every Query limitation has been removed.

The main features developped in ASK/3000 are the following :

- Multi data-set FIND, SORT and REPORT
- Extended FIND possibilities (selection of records on the value of any character of an item ...)
- General sub-item handling; in all commands, a sub-item is used like an item.
- Report generator :
  - Conditional statements; a new statement "IF" enables the user to manage flags conditioning all report statements
  - Call to user-procedures; any problem unsolved by ASK can be solved through a user-procedure called in the report
  - New report statements : Trailer, Nosplit-group, Round etc...
  - Extension of standard statements; for example, 10 Group and Total lines per group level
  - Use of registers in DETAIL, GROUP, TOTAL statements to perform any calculation
- Call of ASK from a user-program; ASK's functions can be used from a user-program by creating an ASK process and sending it an XEQ-file to be executed.
- Etc....

Presentation Abstract

Presentation Title: A DIRECT MAIL ADDRESSES APPLICATION IN A TRANSACTION PROCESSING ENVIRONMENT WITH SHORT RESPONSE TIME REQUIREMENTS.

Author(s): TOIVO TAGEL

Title(s): Director Customer Services

Address: DAFA, FACK
S-10026 STOCKHOLM,
SWEDEN

Phone (08) 738.50.05

Approximate length of your presentation: < 20 ☐ 20 - 40 ☒ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

Please, refer to attached notes.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Abstract: A DIRECT MAIL ADDRESSES APPLICATION IN A TRANSACTION PROCESSING ENVIRONMENT WITH SHORT RESPONSE TIME REQUIREMENTS

By Toivo TAGEL

The Swedish postal authority maintains an independent business division (Postens Adressregister, PAR) involved in providing addresses of Swedish companies and organizations for direct mail purposes. For this division a direct mail addresses data base complete with application programs was developed on a HP 3000 III.

The addresses of the data base may be selected and retrieved in a real time dialogue using logically complex search criteria for qualifying the desired output. The selected addresses may then be printed. The system has been designed to allow a daily output of 100 000 addresses during office hours.

The data base has capacity for 600 000 addresses of firms in addition to 200 000 names of key management staff, e.g. managing directors, financial managers, of those firms. The data base has been designed for this application using a combination of several KSAM and MPE files.

Each address of the data base may be described by a number of search items such telephone area code, postal number, number of employees, annual business turn over as well as an arbitrary number of selection codes describing the line of business of the company. Thus, it is possible to retrieve e. g. all financial managers of firms with a minimum annual turnover, in a certain area and belonging to a specified series of business lines.

The presentation will include a description of the application area and the problems of earlier ADP systems which led to this new development. The particular user environment and the requirements of the application will be discussed.

The very short development time which was available required careful and thorough planning to meet the stated implementation schedule. The planning methods which were used will be mentioned.

The system will be discussed in detail as to its different subsystems for address selection and retrieval, for output of addresses, for data base revision and for updating. A particular emphasis is put on the data management methods and the files design and search logic. The internal design of the system is very much a result of the very short response time requirements for the selection dialogues.

Finally, the application cost justification will be discussed as well as the expected and true impact on the user organization. The good experience of user involvement in the implementation planning will be discussed. The experiences of regular use of the system will be reported. The future development plans will be mentioned.

Presentation Abstract

Presentation Title: DATA COMMUNICATION WITH THE HP 3000 IN EUROPE

Author(s): Session Leader : HEINZ JURACK

Title(s):

Address: HP Böblingen

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The session will begin by reviewing the available datacommunication facilities on the HP 3000.

These will be examined to understand their application in a European environment, which presents special problems due to the differences in different countries.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: PRIVATE VOLUMES AND USER LOGGING

Author(s): Session Leader : KEN SPALDING

Title(s):

Address: HP GSD R&D Group Cupertino

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

This will consist of a two part discussion on :

1 - How to use Private Volumes and Serial Discs. Included are in depth discussions of the use of the VINIT Subsystems, NEWACCOUNT, NEWGROUP, NEWSET, ALTACCOUNT, ALTGROUP, ALTVSET and other commands related to Private Volumes. Stressed will be helpful hints and warings of pitfalls

2 - User Logging, why use User Logging ? How to use the User Logging Intrinsics, how to set up the system for User Logging and how to do recovery under User Logging.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: HP 300 - PRODUCT EVOLUTION AND DIRECTION

Author(s): Session Leader : WALTER UTZ

Title(s):

Address: HP GSD Cupertino

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The traditional use of business computer systems has been centered in EDP departments. The HP 300 is centered in the office environment where the operating manager requires a business decision making tool. Key system design criteria were oriented to normal office environments. The system interacts with business employees in daily operations without requiring specialized EDP knowledge or skills. The local manager controls the operation of the system to serve his need according to his priorities. All important decision making information is available within the department whenever it is needed.

The session explores the concept of the departmental computer system, and its benefits to a wide range of users. The evolution of the system architecture and the design goals to produce an office system are outlined. The direction of the system implementation and its integration into departmental processing are discussed. The session concludes with a projection of the HP 300 design criteria to the business data processing requirements of the 1980s.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Presentation Abstract

Presentation Title: OUTLINE OF PROPOSED PRESENTATION FOR HP 3000 INTERNATIONAL USERS GROUP MEETING IN LYON, FRANCE

Author(s): JOHN STARK

Title(s):

Address: HP

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ **(minutes)**

Abstract: **(No more than 200 words)**

See attached notes, please.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be required by 1 June 1979

Abstract: OUTLINE OF PROPOSED PRESENTATION FOR HP 3000 INTERNATIONAL USERS GROUP MEETING IN LYON, FRANCE

By John STARK

I. TOPIC

Interfacing Quantitative Modeling Techniques to MFG 3000

II. BACKGROUND

A - MFG 3000

1. What does it do ?
2. What doesn't it do ?

B - Some useful mathematical programming models

1. Exponential Smoothing
    - a) What is it ?
2. Net Present Value
    - b) What is it ?
3. Linear Programming
    - a) Single period
        - 1- What is it ?
        - 2- What are its limitations ?
    - b) Multi period
        - 1- What is it ?
        - 2- What are the advantages over single period ?

III. APPLICATION OF MODELS and MFG 3000

A - Exponential Smoothing

1. Forecast Demand
    - a) Establishes a master schedule of end items

B - Multi period Linear Programming / Net Present Value

1. Determines if forecasted demand will maximize net present value of return on investment to firm
    - a) Adjust master schedule to maximize net present value
        - 1- Realize that some opportunities cannot be carried over to subsequent periods
            - o Implies a sub- optimum solution may be necessary

C - MRP 3000

1. Determines lower level requirements to build master scheduled items.

D - Multi Period Linear Programming

1. Utilize to optimize mix of detail parts to be fabricated

a) Capacity constraints

IV. EXAMPLE

A - Happy Peddler Bicycle Company

Presentation Abstract

Presentation Title: WORKSHOP FOR HP 300 OEM's

Author(s): Session Leader : DAVE IUPPA

Title(s):

Address: HP Böblingen

Phone

Approximate length of your presentation: < 20 ☐ 20 - 40 ☐ > 40 ☒ (minutes)

Abstract: (No more than 200 words)

The session will provide an opportunity for existing and interested OEM's to exchange information on the HP 300 product line.

Topics expected to be covered include Applications currently planned, or being implemented, contributed Library usefulness, utilisation of special features in customers applications.

N. B. Product Line Evolution is the subject of a separate session.

Please return this form to: John Eaton
London Business School
Sussex Place
Regent's Park
LONDON NW1 4SA England

by Friday 20 April

Final papers will be [illegible]

PROCEEDINGS

PROCESS MANAGEMENT IN MPE-III

CHRIS MOELLER
HP CUPERTINO.

- CONTENTS -

1. USER'S VIEW OF PROCESSES
2. DEFINITION OF PROCESS IN MPE
3. PROCESS STATES
4. PROCESS ACTIVITIES
5. PROCESS SCHEDULING
6. SIMPLE DISPATCHING
7. COMPLICATED DISPATCHING

USER VIEW OF PROCESS

- AN ENTITY IN MPE THAT WILL ALLOW A USER'S CODE TO BE EXECUTED.

- SOMETHING THAT REQUIRES PH CAPABILITY TO USE AND HAS SOME NICE INTRINSICS ASSOCIATED WITH IT.

  - CREATE, KILL, TERMINATE
  - ACTIVATE, SUSPEND
  - GETPROCID, GETPROCINFO, GETORIGIN, FATHER
  - MAIL, SENDMAIL, RECEIVEMAIL
  - GETPRIORITY

PROCESS DEFINITION

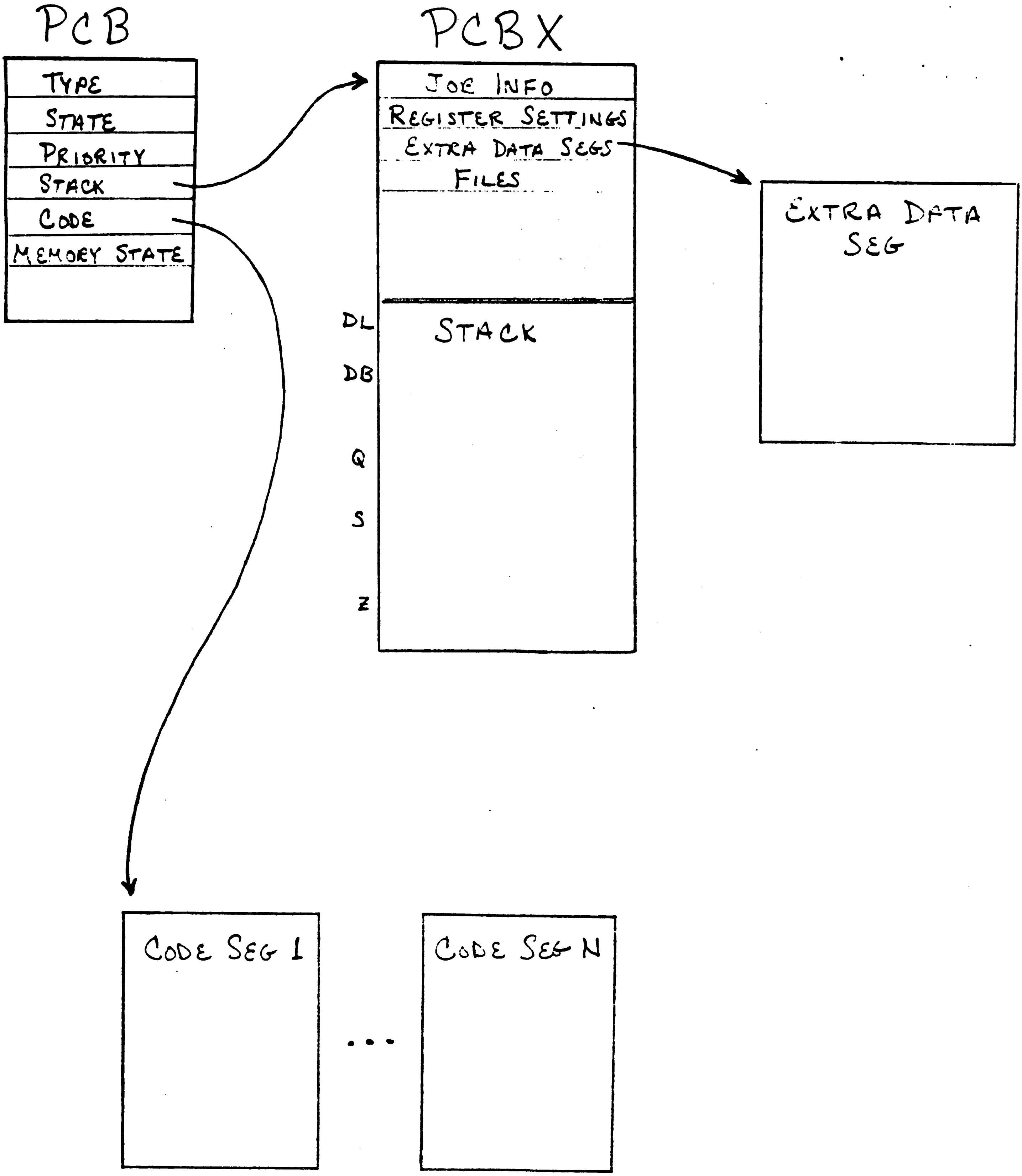
A PROCESS IS THE ENTITY IN MPE WHICH CAN ACCOMPLISH WORK. IT IS THE ENTITY WHICH ALLOWS SYSTEM RESOURCES TO BE SHARED.

A PROCESS MUST ALWAYS HAVE

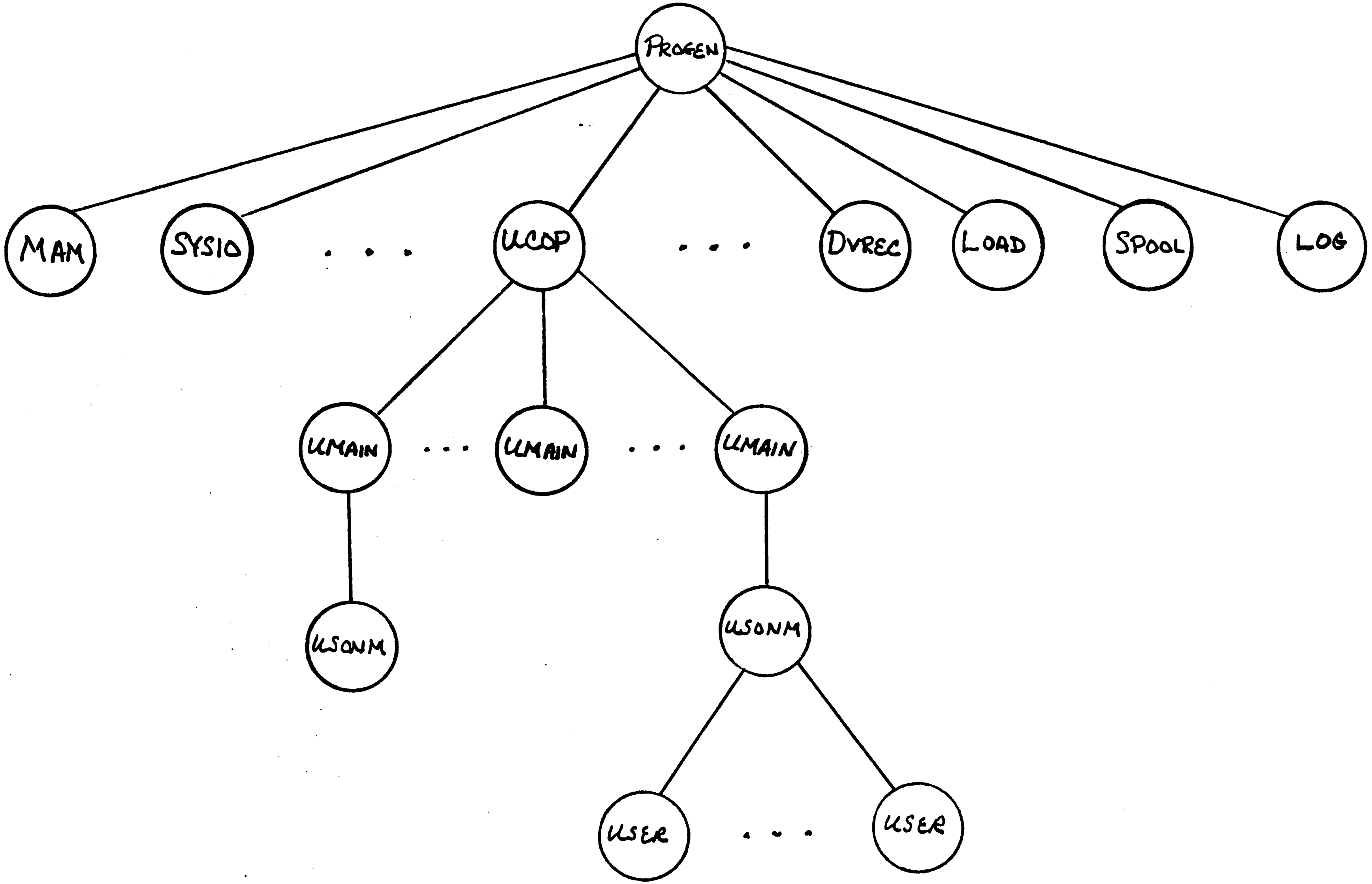
- A CODE SPACE
- A DATA SPACE
  - o PRIVATE, UNSHAREABLE (STACK)
  - o POSSIBLY SHAREABLE (EXTRA DATA SEGS)
- CONTROL STRUCTURES
  - o PCB (CORE RESIDENT)
  - o PCBX (NON-RESIDENT)

A PROCESS MAY HAVE OTHER RESOURCES AS WELL (RINS, FILE/DEVICES, OTHER PROCESSES)

STRUCTURE OF A PROCESS


PCB
TYPE
STATE
PRIORITY
STACK
CODE
MEMORY STATE
PCBX
JOB INFO
REGISTER SETTINGS
EXTRA DATA SEGS
FILES
DL
STACK
DB
Q
S
Z
EXTRA DATA SEG
CODE SEG 1
...
CODE SEG N

PROCESS HIERARCHY IN MPE-III


PROGEN
MAM
SYSIO
. . .
UCOP
. . .
DVREC
LOAD
SPOOL
LOG
UMAIN
. . .
UMAIN
. . .
UMAIN
USONM
USONM
USER
. . .
USER

PROCESS STATE

- MAJOR STATE

o ACTIVE	PROCESS IS RUNNING OR READY TO RUN
o WAIT	SOME EVENT MUST OCCUR BEFORE PROCESS CAN RUN

- ADDITIONAL MODES

o "SHORT" WAIT	DISC TRANSFER
o "LONG" WAIT	I/O, FATHER, SON, RIN, MAIL, MOURNING, JUNK
o IMPEDED	REALLY JUST A WAIT, BUT HOPEFULLY SHORTER
o CRITICAL	PROCESS MUST NOT BE ABORTED
o SIR	PROCESS HAS A "SYSTEM INTERNAL RESOURCE"

PROCESS ACTIVITIES

IN THE COURSE OF ITS EXISTENCE, A PROCESS UNDERGOES SEVERAL ACTIVITIES

- o CREATION
- o TERMINATION
- o COMMUNICATION
- o SYNCHRONIZATION

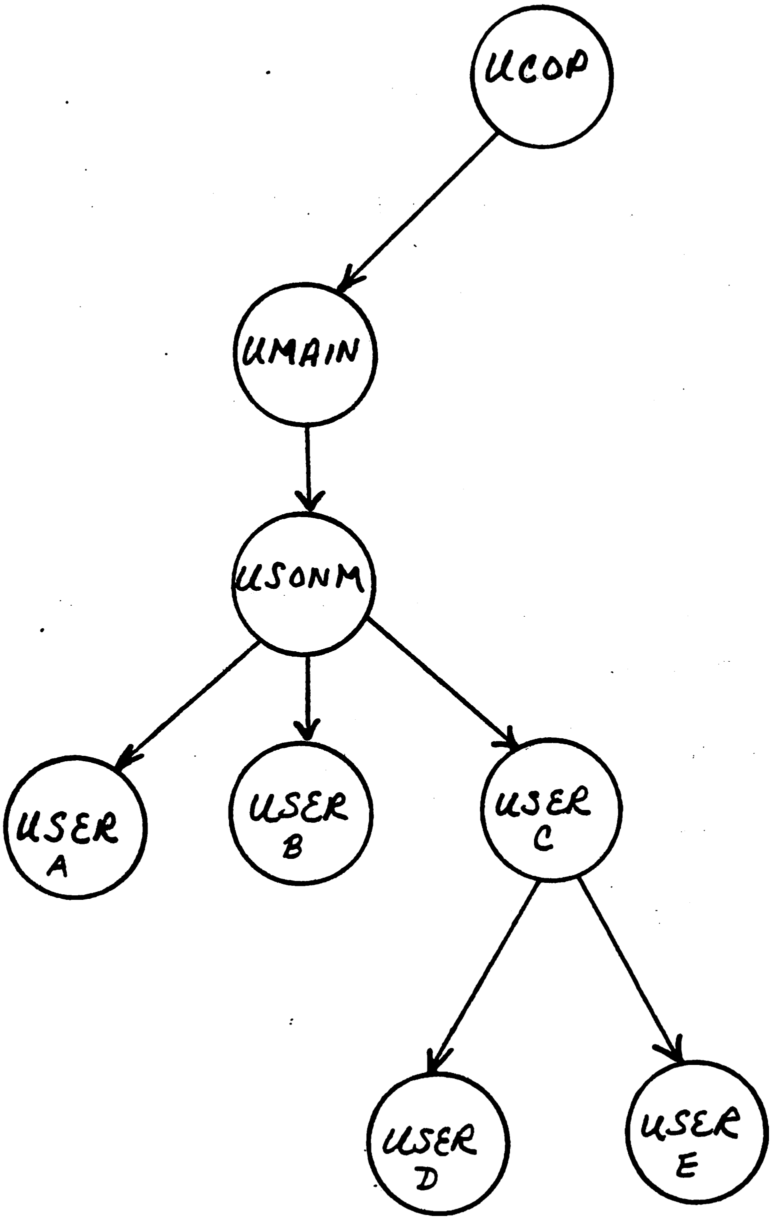
PROCESS CREATION

- GET A PCB FOR THE NEW PROCESS
- LINK PROGRAM FILE IN VIRTUAL MEMORY
- GET A DAT SEGMENT FOR THE NEW STACK
- FORMAT THE PCBX
- FORMAT PCB AND LINK INTO STRUCTURE
- MARK PCB AS BEING IN FATHER WAIT

PROCESS TERMINATION

- GENERAL PHILOSOPHY: LET A PROCESS GIVE UP AS MANY OF ITS RESOURCES AS POSSIBLE.
- HAVE SOME OTHER PROCESS REMOVE STACK AND PCB OF TERMINATING PROCESS.
- ALGORITHM:
  - o LET PROCESS RETURN "NON-ESSENTIAL" RESOURCES TO SYSTEM (RINS, FILES, ETC.)
  - o "STOP" SONS FROM RUNNING AND FORCE THEM TO START TERMINATE SEQUENCE.
  - o WAIT FOR EACH SON TO "DIE" AND THEN RETURN ITS STACK AND PCB.
  - o ASK FATHER OR UCOP TO TAKE AWAY THIS STACK AND PCB.

PROCESS DELETION EXAMPLE


UCOP
UMAIN
USONM
USER A
USER B
USER C
USER D
USER E

PROCESS COMMUNICATION

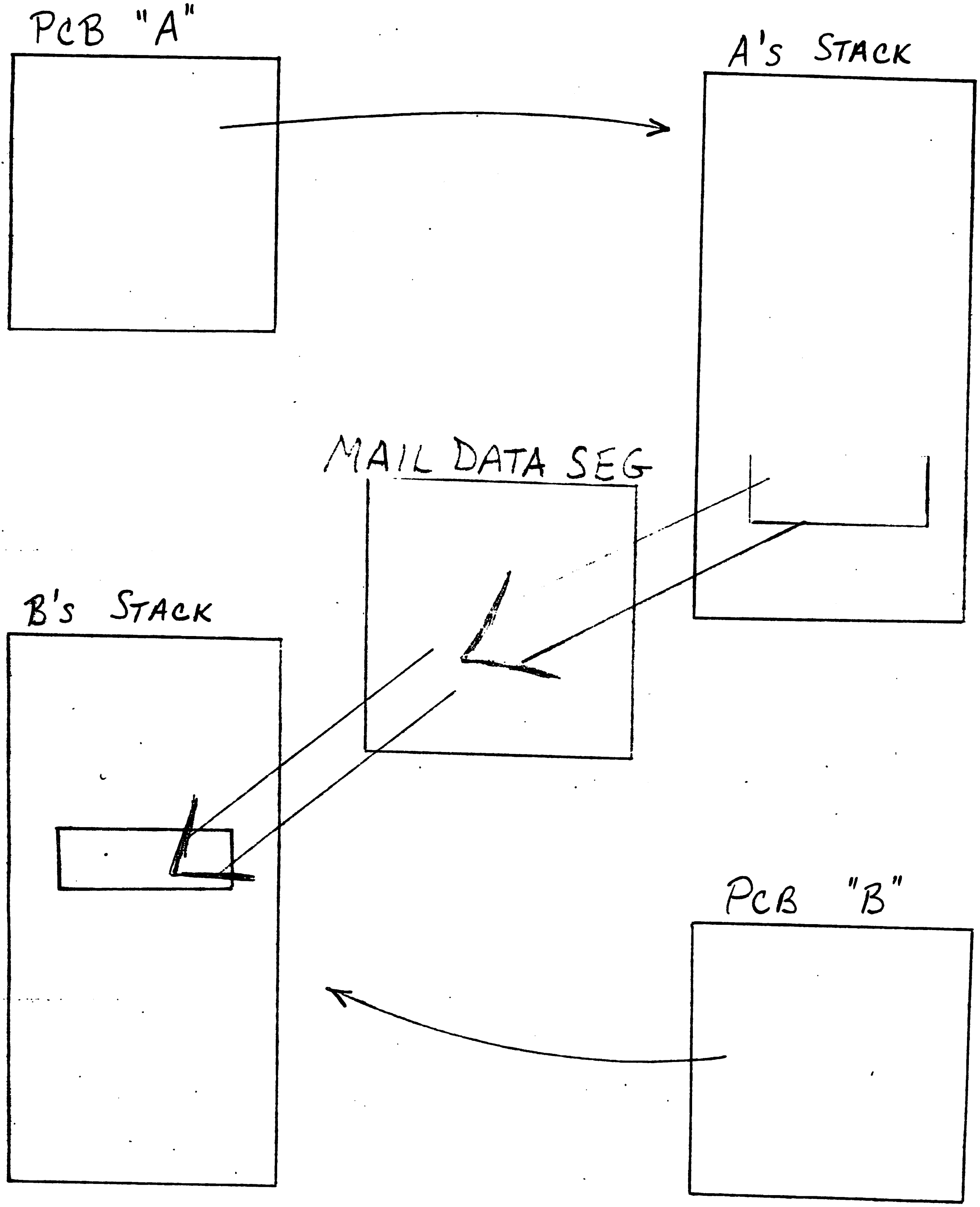
- A PROCESS MAY COMMUNICATE WITH OTHER PROCESSES THROUGH A VARIETY OF MEANS.
  - o SHARED FILE
  - o EXTRA DATA SEGMENT
  - o MAIL FACILITY

- A PROCESS MAY COMMUNICATE WITH ITS FATHER OR 1 OF ITS SONS THROUGH A "MAILBOX".

- BETWEEN ANY TWO PROCESSES, THERE CAN BE ONLY 1 OUTSTANDING MESSAGE AT A TIME.

- MAIL MUST BE "SENT" AND "RECEIVED".

MAIL MECHANISM

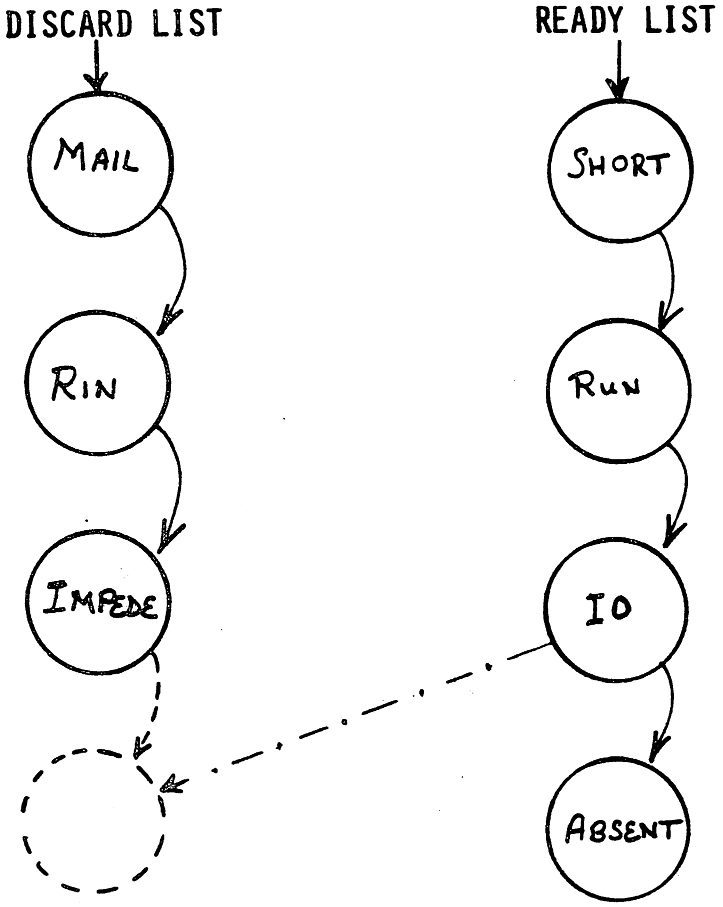

PCB "A"
A's STACK
MAIL DATA SEG
B's STACK
PCB "B"

PROCESS SYNCHRONIZATION

A PROCESS MAY CAUSE ANOTHER PROCESS TO BECOME ACTIVE OR IT MAY CAUSE ITSELF TO BECOME SUSPENDED.

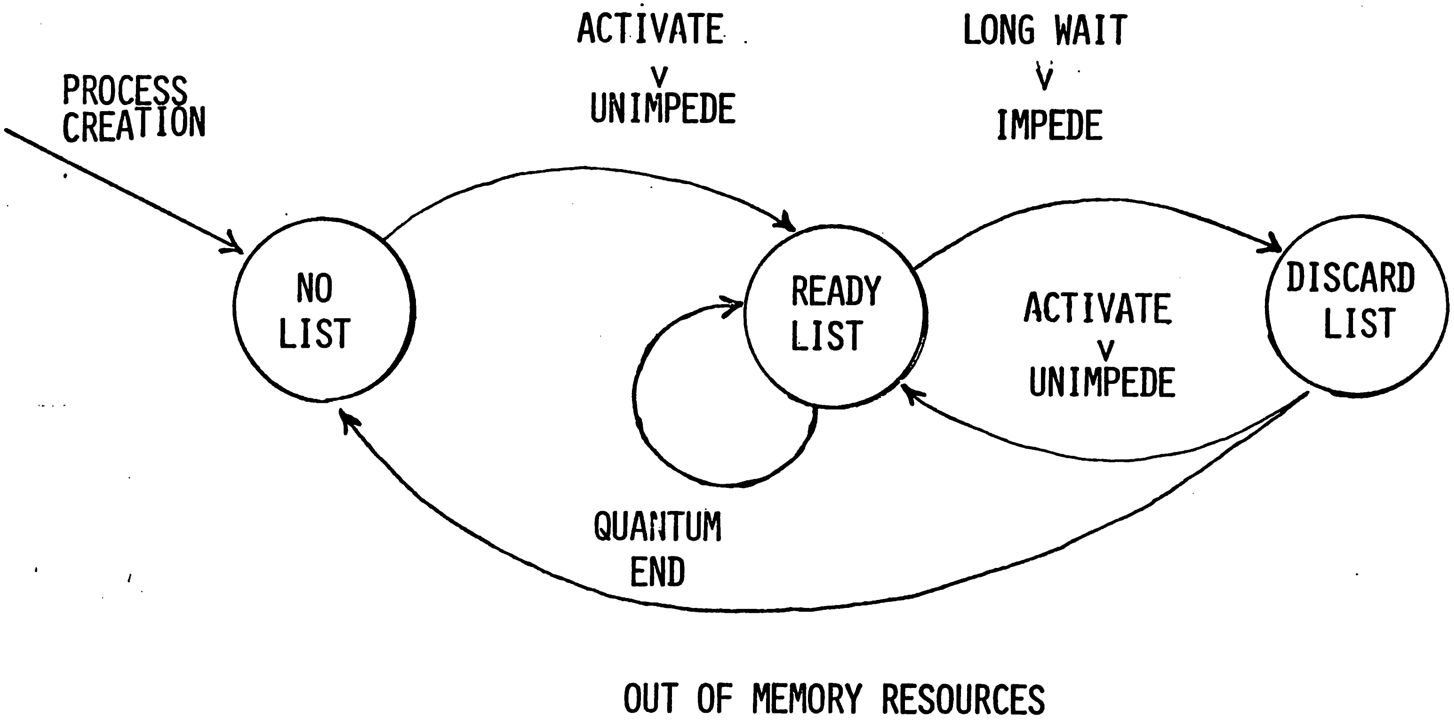
```
PROCEDURE ACTIVATE (P);
   BEGIN
     MARK P'S PCB AS "ACTIVE";
     LINE PCB P INTO READY LIST;
   END;


PROCEDURE SUSPEND;
   BEGIN
     MARK MY PCB AS "WAITING FOR EVENT";
     EXECUTE DISP INSTRUCTION;
   END;
```

DISPATCHER'S SCHEDULING QUEUES


DISCARD LIST
MAIL
RIN
IMPEDE
READY LIST
SHORT
RUN
IO
ABSENT

- READY LIST	ALL PROCESSES WHICH WANT TO USE THE CPU
- DISCARD LIST	PROCESSES WHICH CAN'T USE CPU NOW AND HAVE MEMORY.

HOW PROCESSES MOVE BETWEEN LISTS


ACTIVATE
v
UNIMPEDE
LONG WAIT
v
IMPEDE
PROCESS
CREATION
NO
LIST
READY
LIST
ACTIVATE
v
UNIMPEDE
DISCARD
LIST
QUANTUM
END
OUT OF MEMORY RESOURCES

DYNAMIC PRIORITIES AND READY LIST

- PRIORITIES ARE INTEGERS BETWEEN 1 AND 255 (1 IS HIGHEST; 255 IS LOWEST)
- READY LIST IS ORDERED BY PRIORITY AND MANAGED AS A SET OF SUBQUEUES.
- A SUBQUEUE IS A RANGE OF PRIORITIES AND MANAGED LINEARLY OR CIRCULARLY.
- PROCESSES IN A AND B QUEUES HAVE FIXED PRIORITIES.
- PROCESSES IN C, D, AND E QUEUES HAVE DYNAMIC PRIORTIES.

MOVEMENT RULES FOR C, D, AND E

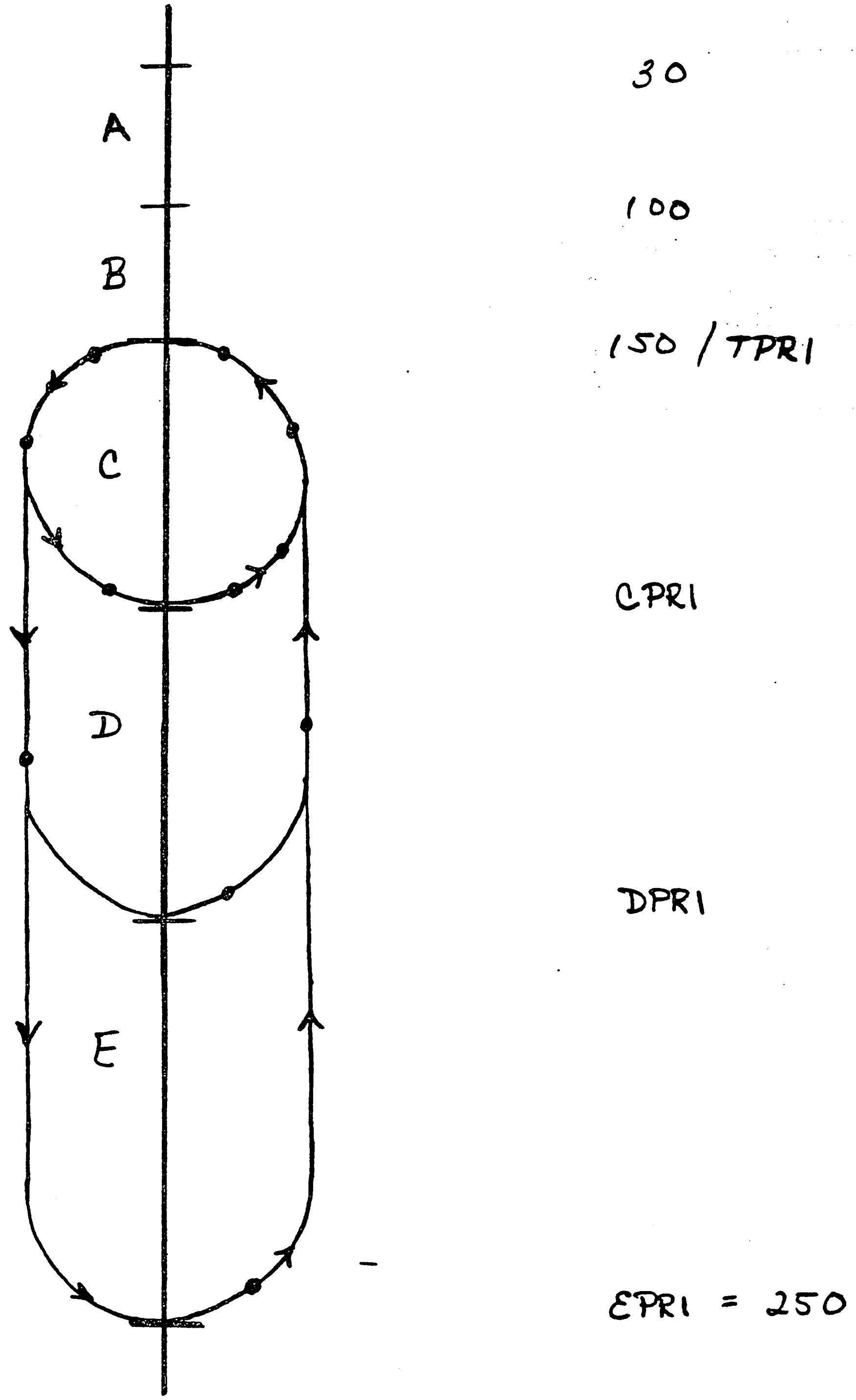
- INCREASE RULE: IF PROCESS X CUTS IN FRONT OF PROCESS Y THEN INCREASE Y'S PRORITY BY 1 (I.E. $PRI_Y := PRI_Y - 1$)
- DECREASE RULE: AFTER COMPLETION OF AN ENTIRE QUANTUM

FOR CS ----> NEWPRI := OLDPRI + 1
(BUT NO LOWER THAN CPR1)

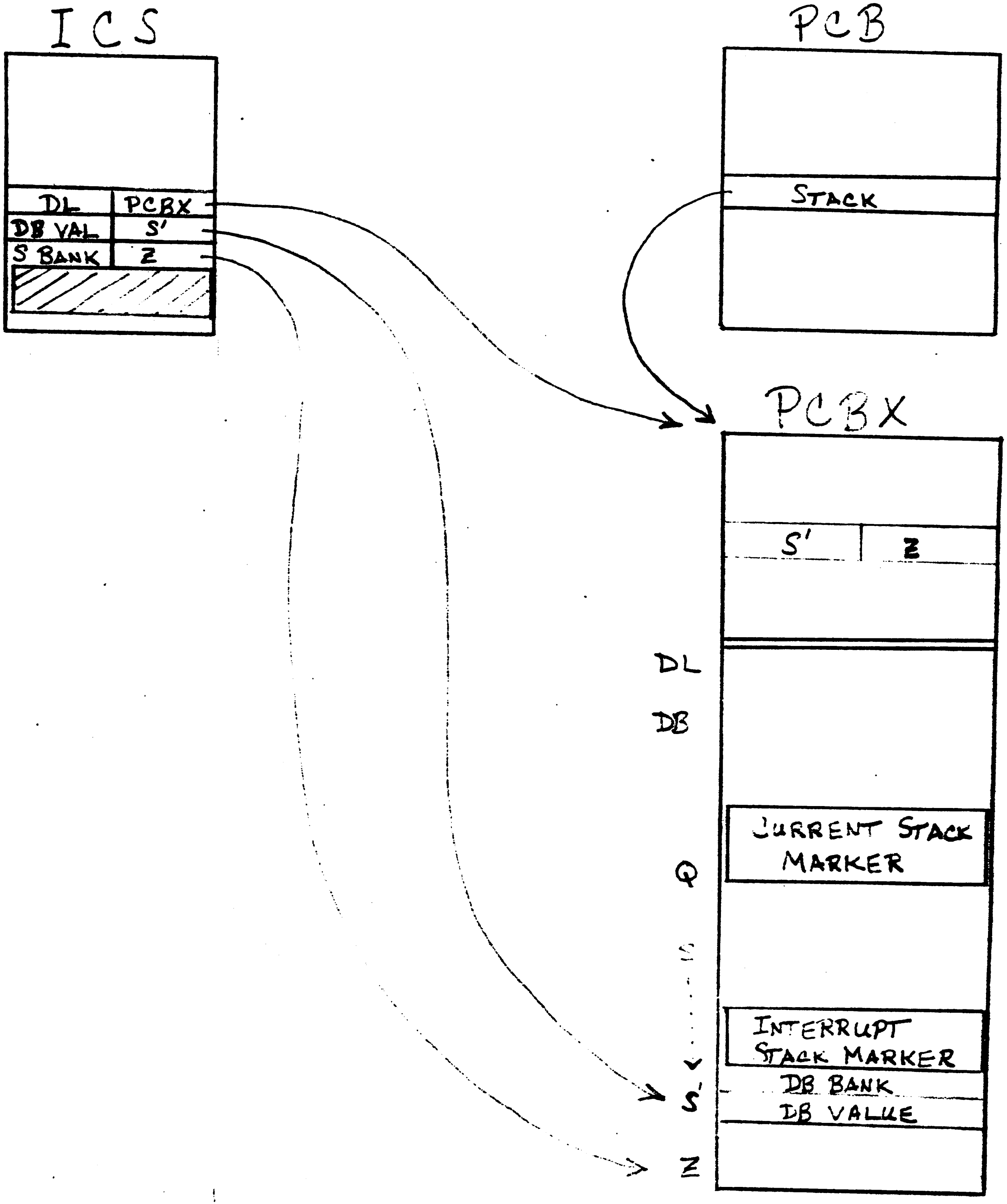
FOR DS ----> NEWPRI := (((OLD PRI - 150) +1) * 3) + 150
(BUT NO LOWER THAN DPRI)

FOR ES ----> NEWPRI := 250 ALWAYS

READY LIST AS SUBQUEUES


30
A
100
B
150 / TPRI
C
CPRI
D
DPRI
E
EPRI = 250

PROCESS QUIESCE


ICS
DL
PCBX
DB VAL
S'
S BANK
Z
PCB
STACK
PCBX
S'
Z
DL
DB
CURRENT STACK MARKER
Q
S
INTERRUPT STACK MARKER
S'
DB BANK
DB VALUE
Z

SIMPLE DISPATCHING

DISPATCHER IS THE SYSTEM ROUTINE WHICH DETERMINES THE FAIR ALLOCATION OF THE CPU AMONG THE CURRENTLY ACTIVE PROCESSES.

```
PROCEDURE DISPATCHER;
  BEGIN
    << QUIESCE THE RUNNING PROCESS >>
    UPDATE S-REGISTER SETTING IN PCBX;
    IF USER-LEVEL PROCESS THEN UPDATE CPU USE;
    IF CIRCULARLY-SCHEDULED AND QUANTUM OVER THEN
       COMPUTE NEW PRIORITY;

    << LAUNCH THE NEXT READY PROCESS >>
    SCAN READY LIST FOR HIGHEST PRIORITY PROCESS;
    READ REGISTER SETTINGS PCBX ----> ICS;
    SET TIMER WITH PROCESS'S QUANTUM;
    IXIT TO START PROCESS EXECUTION;
  END;
```

COMPLICATED DISPATCHING

- HIGHEST PRIORITY, READY PROCESS MAY NOT HAVE ALL REQUIRED SEGMENTS PRESENT IN MAIN MEMORY.

- DISPATCHER DETECTS THE PROCESS IS OUT OR ABSENT AND SCHEDULES MEMORY MANAGER FOR EXECUTION.

- MAM EXECUTES, SATISFYING THE REQUEST
  - o FIND REAL MEMORY SPACE
  - o INITIATE READ AND SHORT WAIT
  - o COMPLETE REQUEST; MARK PROCESS INCORE

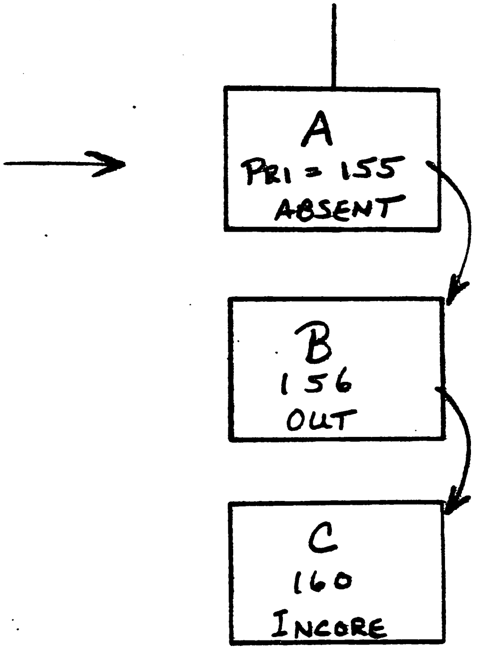
- DISPATCHER NOW FINDS HIGH PRIORITY PROCESS INCORE AND LETS IT RUN.

- WHILE MAM IS WAITING/WORKING, THE NEXT HIGHEST PRIORITY, INCORE PROCESS WILL BE RUN.

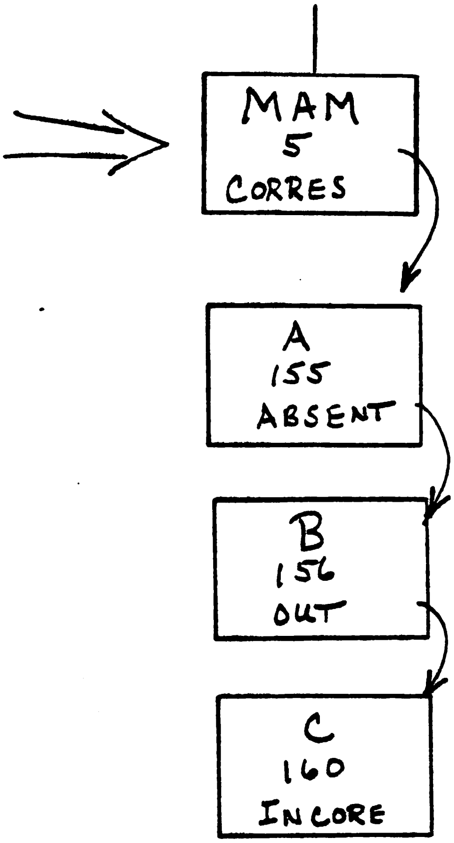
- SIDE NOTE: IF HIGHEST PRIORITY PROCESS EVER SHORT WAITS, ONLY INCORE LOWER PRIORITY PROCESSES MAY BE RUN.

DISPATCHING EXAMPLE

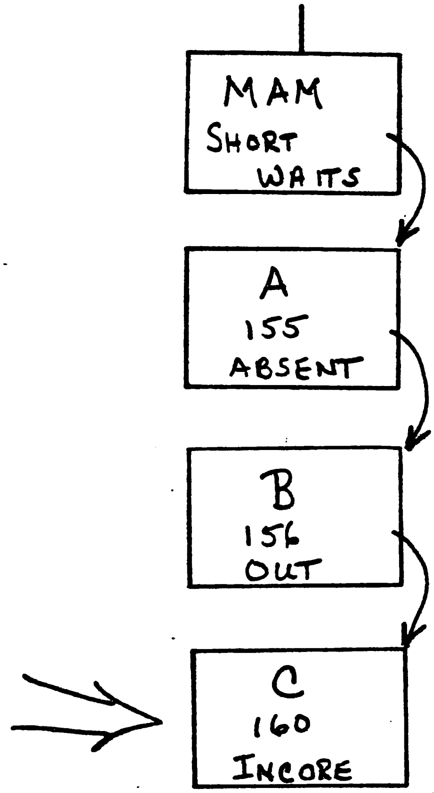
1. READY LIST


A
PRI = 155
ABSENT
B
156
OUT
C
160
INCORE

2. READY LIST

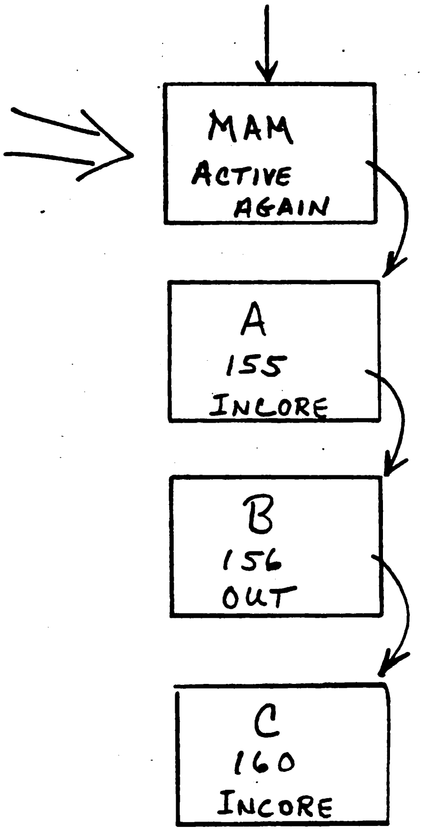

MAM
5
CORRES
A
155
ABSENT
B
156
OUT
C
160
INCORE

3. READY LIST

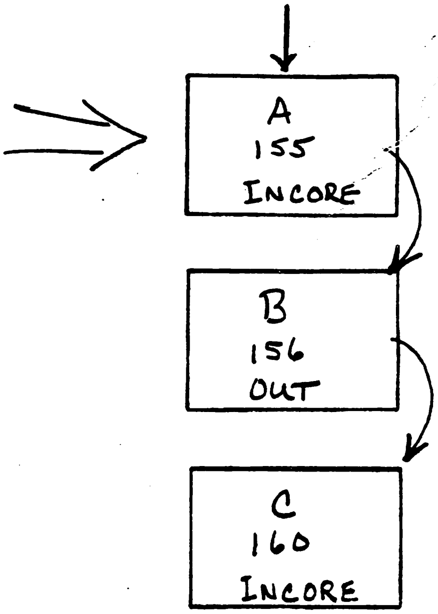

MAM
SHORT
WAITS
A
155
ABSENT
B
156
OUT
C
160
INCORE

DISPATCHING EXAMPLE

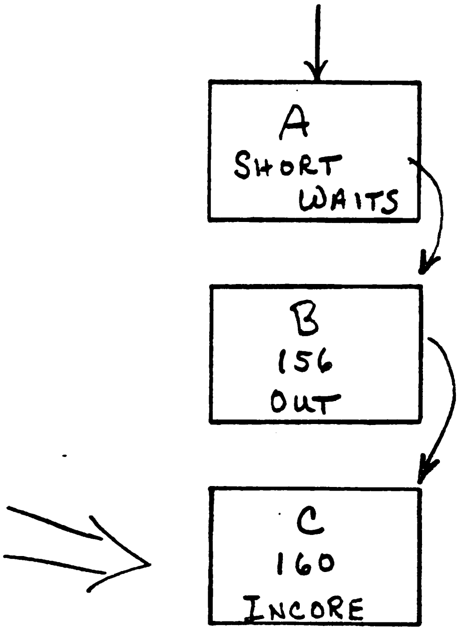
4. READY LIST


MAM
ACTIVE AGAIN
A
155
INCORE
B
156
OUT
C
160
INCORE

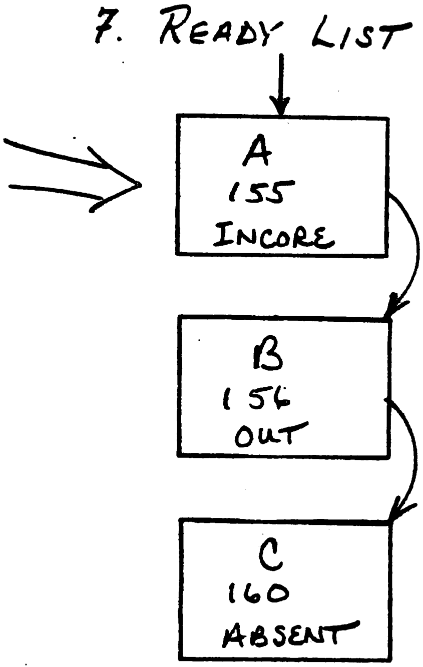
5. READY LIST

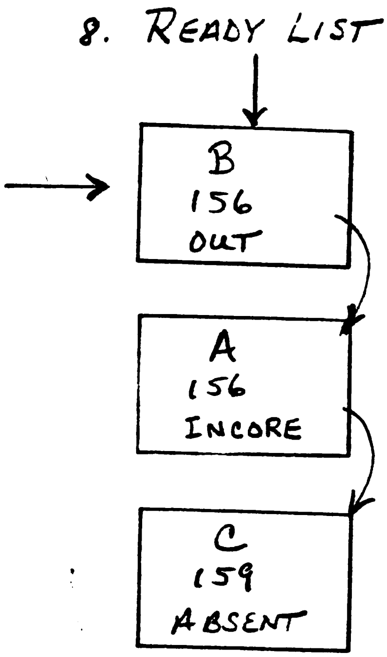

A
155
INCORE
B
156
OUT
C
160
INCORE

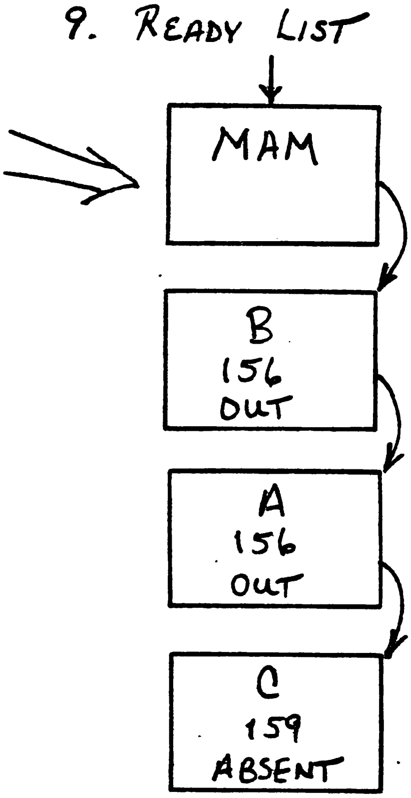
6. READY LIST

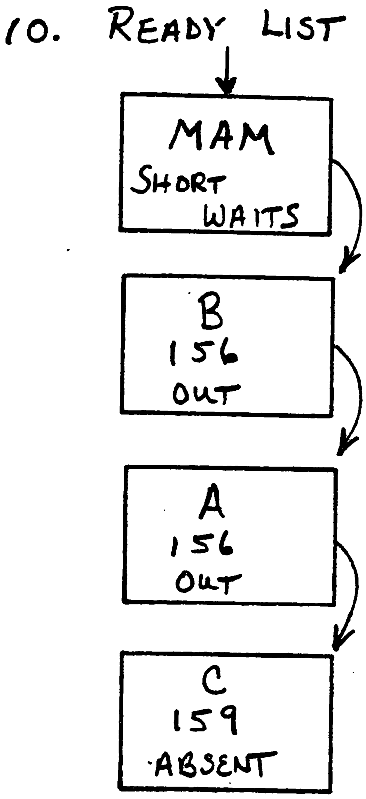

A
SHORT WAITS
B
156
OUT
C
160
INCORE

DISPATCHING EXAMPLE


7. READY LIST
A
155
INCORE
B
156
OUT
C
160
ABSENT

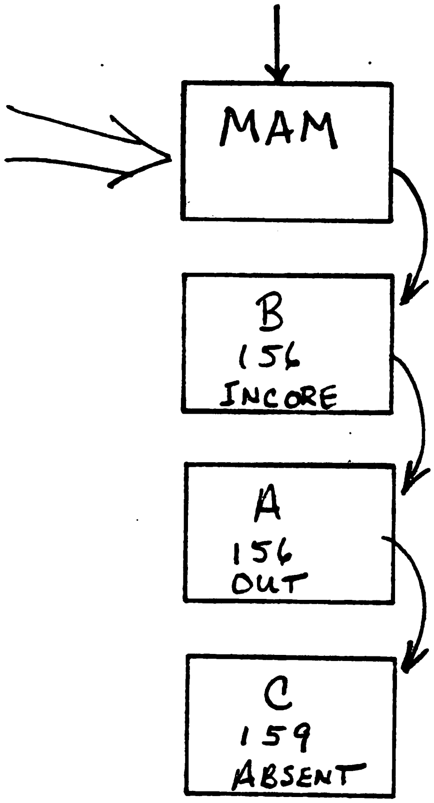

8. READY LIST
B
156
OUT
A
156
INCORE
C
159
ABSENT


9. READY LIST
MAM
B
156
OUT
A
156
OUT
C
159
ABSENT

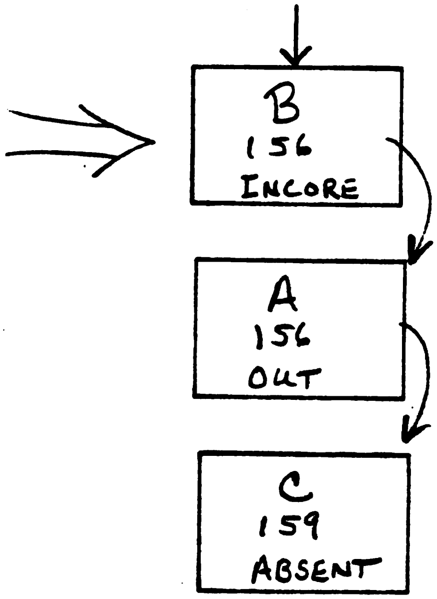

10. READY LIST
MAM
SHORT
WAITS
B
156
OUT
A
156
OUT
C
159
ABSENT

DISPATCHING EXAMPLE

11. READY LIST


MAM
B
156
INCORE
A
156
OUT
C
159
ABSENT

12 READY LIST


B
156
INCORE
A
156
OUT
C
159
ABSENT

AN INTEGRATED SYSTEM FOR DOCUMENTATION MAINTENANCE, PROGRAM AND JOB SELECTION, AND MEMO PROCESSING

by

John A. Schick

Wm C. Brown Company, Publishers
Dubuque, Iowa

INTRODUCTION

One important requirement of any application system is the ability of the end user to conveniently obtain access to the various application programs. This is especially true when the end user lacks sophistication in the use of computer systems. User frustration and the possible loss of data integrity are the likely results of a system that requires novice users to directly run programs and stream jobs with cryptic file names qualified by group and account. One rudimentary solution is to use mnemonic names for program and job files which reside in the logon groups and accounts of the various users. This solution is less than completely satisfactory for several reasons:

i) Although mnemonic file names are by definition fairly easy to remember, users must still remember exact names.

ii) The eight character file name limit imposed by the operating system makes a consistent system of mnemonic names difficult to devise.

iii) To eliminate the need for group and account qualification, multiple copies of program and job files must be maintained if they are to be used from more than one group or account.

iv) Mnemonic names do not readily lend themselves to the classification of programs by system, the cross-referencing of data files to programs, and other similar aspects of program documentation important to the system maintenance function.

The Wm. C. Brown Company used mnemonic file names for about one year until the volume of systems and programs made the drawbacks of this technique evident. This paper describes the system developed to improve the friendliness of the program and job selection process and at the same time allow a systematic naming convention for all program, job, and data files. This system consists of a data base-driven process handling program (SELECTOR) which runs application programs and streams jobs for the end user, plus a set of documentation maintenance and reporting programs which maintain and document the data base used by SELECTOR as well as the actual source, USL, program, and documentation text files. Also discussed are two spinoffs of the original system: a job streaming utility (STREAMER) and a memo processing subsystem. Figure 1 shows a generalized system flowchart of the entire system.

The Program SELECTOR

SELECTOR is a process handling program written in recognition of the need for an interface between the end user and MPE. Through SELECTOR, users can run programs and stream jobs either by entering mnemonic names or by "X"-ing descriptions displayed according to a pre-defined network of menus. SELECTOR is the only program that need be run by most of our operators. Indeed, since UDC's became available they are prevented from running any other programs and do not even need to explicitly run SELECTOR because of the following UDC in effect for most users:

```
SELECTOR
OPTION LOGON,NOBREAK
CONTINUE
RUN SELECTOR.PUB.SYS
BYE
```

In several respects, SELECTOR simulates MPE. The program begins in character mode by displaying a double colon prompt. In response to this prompt, the user may enter any of the following:

i) any of a limited number of "safe" MPE commands which are carried out by means of the COMMAND intrinsic

ii) the mnemonic name (up to twenty characters) of a main program to be run or a job to be streamed

iii) the special mnemonic "HELP" which causes SELECTOR to enter block mode where menus of system and program descriptions are displayed for the user to choose from in "X-marks-the-spot" fashion

iv) the special mnemonics "BYE", "END", or "EXIT", all of which terminate SELECTOR.

SELECTOR employs a documentation data base named BASE03 to make the correspondence between the operator's input and the actual main program or job to be invoked. Figure 2 presents the schema layout of the most important data sets of BASE03. Each main program, subprogram, and job on our system is identified by an entry in the PROGRAM data set. The twenty character field MNEMONIC-NAME provides the means of associating mnemonic program names with actual internal file names. When an operator of SELECTOR enters a mnemonic name, the PROGRAM set entry containing that mnemonic name is retrieved and the corresponding file name is constructed from the fields PROG-ID, GROUP-NAME, and ACCOUNT-NAME. If the PROG-TYPE field contains "MAIN" (indicating a main program), the constructed file name is created and activated. If the PROG-TYPE field contains "JOB" (indicating a job file), the constructed file name is streamed.

In "HELP" mode, SELECTOR uses the contents of the data sets SELECTORS and OPTIONS to determine the nature of the menu network. Each menu is defined by an entry in the SELECTORS set containing a forty character SELECTOR-TITLE field to be displayed at the top of the menu and a two character SELECTOR-CODE. The individual options within each menu are defined by means of OPTIONS set entries containing the SELECTOR-CODE of the menu on which they are to appear. An option that leads to another menu contains the SELECTOR-CODE of the desired next menu in its NEXT-SELECTOR field. An option that runs a program or streams a job when it is selected is linked to the corresponding PROGRAM set entry via its PROG-ID field. A control set in BASE03 contains the SELECTOR-CODE of the first menu to be displayed whenever "HELP" mode is entered, as well as the name of the VIEW forms file containing the form on which the menus are to be displayed. Figure 3 shows the first menu that is displayed at our installation when SELECTOR is run in "HELP" mode.

The Documentation Maintenance Subsystem

The method used to name files can provide important documentation information to programmers and operations personnel. Since SELECTOR has eliminated any need to name files mnemonically, we have adopted a naming convention at the Wm. C. Brown Company that communicates the following information about each file:

i) the file type

ii) except for data bases, the system to which the file belongs

iii) in the case of data files, the program or job which makes principal use of the file.

Data bases, which frequently cross system boundaries, are identified by the four character code "BASE" followed by a two digit sequence number assigned sequentially from 01. Data bases designed before the adoption of this convention (such as the invoicing data base INVCS) still stubbornly retain their old names but all new data bases adhere to this standard.

All other files are identified by a four character system identification code followed by a one character file type code and a three digit sequence number. The following is a list of the various file type codes currently in use:

C = code of all kinds (source, USL, program, and documentation files)
D = data files not included in any of the other categories
F = forms files
J = job files
K = KSAM key files
L = KSAM data files
P = procedure documentation files
R = report files
S = system documentation files
T = miscellaneous text files
W = work files
Y = copylib files.

All the files associated with the installation of a program and the production of program documentation are assigned the same name within separate groups of a standard account (WCBLIB) via the type code C. For example all source files are found in the group SOURCE.WCBLIB and all text files containing operating instructions are contained in the group INSTRUCT.WCBLIB. When source, program, and USL files must all reside in the same group (e.g. in the programmer's logon group during program development), type codes U for USL files and O for program (object) files are used. Sequence numbers for the programs within each system are normally assigned from 010 in multiples of ten.

Data and job files are numbered to cross-reference them to programs. Data files are assigned the sequence number of the first (and frequently only) program that uses the file. Jobs are numbered according to the first program run in that job. The rare job that contains no run commands is assigned a sequence number in its own right.

This naming convention has served well as a program classification and cross-referencing aid. It also has allowed for a systematic process of program installation and documentation production.

As mentioned earlier, documentation on each main program, subprogram, and job is maintained in the data base BASE03. In addition to the header information in the PROGRAM set, information on data files and copylibs is maintained in the sets DATA-FILE and COPYLIB respectively. The sets PROG-PROG-XREF, PROG-FILE-XREF, and PROG-COPY-XREF contain the program, data file, and copylib cross-reference data that their names imply.

Besides the fixed-format information in BASE03, editor files are used to maintain the free-format aspects of program documentation. These files contain general program descriptions, revision histories, and operating instructions which reside in the DOCTEXT, FIXLIST, and INSTRUCT groups of the WCBLIB account.

A documentation maintenance program (DOCUC060) using VIEW, handles most of the aspects of maintaining BASE03, maintaining the program, USL, source, and text files in WCBLIB, and printing program documentation. As can be seen in figure 4, DOCUC060 offers eleven separate options:

i) Install a program into WCBLIB (optionally FCOPYing the source from any specified file) without also compiling all called or calling programs.

ii) Install a program including the compilation of all called or calling programs.

iii) Print program documentation on any specified device class or device number.

iv) Change a program's header information (in the PROGRAM data set).

v) Change a program's file usage information (in the DATA-FILE and PROG-FILE-XREF sets).

vi) Change a program's list of called programs (in the PROG-PROG-XREF set).

vii) Change a COBOL program's copylib usage information (in the COPYLIB and PROG-COPY-XREF sets).

viii) Add header, file usage, program usage, and copylib usage data for a new program.

ix) Delete all information for an obsolete program.

x) Change the name of an existing program.

xi) Invoke EDITOR to maintain the documentation text files in the DOCTEXT, FIXLIST, and INSTRUCT groups.

Figures 5 through 8 show the screens that are displayed during the process of adding documentation for a new program via DOCUC060. Figure 9 is a sample first page from program documentation produced by DOCUC060.

DOCUC060 does not maintain the SELECTORS and OPTIONS data sets used by SELECTOR. A separate program to maintain these sets is on the drawing board. They are currently maintained through QUERY.

The Job Streaming Utility STREAMER

It should come as no great revelation that many of the functions of an application system are best performed in batch jobs. For the sake of operator convenience, the commands for all such jobs should reside in job files requiring no maintenance by the user. The MPE stream facility, however, has several fundamental limitations that precludes the use of standard job files. !JOB statements must be hard-coded with user and account names and passwords which represents a potential accounting and security problem and a tedious maintenance problem whenever passwords are changed. Also, the stream facility makes no provision for the dynamic entry of variable data into job streams at stream time. To overcome these limitations, a job streaming utility named STREAMER has been written to stream all jobs at our installation.

STREAMER reads a file that has been equated to the formal file designator JOBFILE ($STDIN by default), inserts the user's logon name, group, and/or account if any of these parameters are missing in the !JOB statement, inserts passwords if they are omitted, inserts data solicited from the operator in response to input commands in the job file, and then streams the job using the COMMAND intrinsic. STREAMER runs in privileged mode to extract passwords directly from the system directory. For the sake of system security, SELECTOR imposes several restraints to prevent or at least monitor one user streaming a job that names another user in its !JOB statement:

i) System manager jobs can only be streamed by users with system manager capability unless the job file contains the correct passwords.

ii) Account manager jobs can only be streamed by users with system manager capability or account manager capability in the same account unless the job file contains the correct passwords.

iii) System supervisor jobs can only be streamed by users with system manager or system supervisor capability unless the job file contains the correct passwords.

iv) A console message is produced and a comment line is inserted into the job stream identifying the operator if the operator's logon id does not match the job's user and account names.

To allow for entry of variable data into job streams, STREAMER provides for two pseudo-MPE commands !INPUT and !INPUTS. When STREAMER encounters an !INPUT command, it displays the rest of the command on $STDLIST as a prompt for the operator, reads the operator's reply from $STDIN and inserts the reply into the job stream in place of the !INPUT command itself. The !INPUTS command is processed the same way except that the prompt is repeated and replies inserted into the job stream until the operator gives a blank response.

STREAMER was designed to resemble as closely as possible the MPE STREAM command. It is most conveniently run by means of the following UDC:

```
STREAMER JOBFILE=$STDIN,REPLACE=0
FILE JOBFILE=!JOBFILE
RUN STREAMER.PUB.SYS;PARM=!REPLACE
RESET JOBFILE
```

Using this UDC, STREAMER is invoked exactly like the STREAM command with the one exception that the colon replacement character must be expressed numerically since it is communicated to STREAMER via the PARM= parameter of the RUN command.

STREAMER is used both stand-alone and from programs to stream jobs throughout our system. An SPL utility subprogram has been written to issue the file equation for JOBFILE and to create and activate STREAMER so that STREAMER can be invoked from application programs. It was decided to use an SPL interface routine to process-handle STREAMER rather than write a subprogram version of STREAMER so that calling programs would not need privileged mode capability but just process handling capability instead.

The Memo Processing Subsystem

An unanticipated spinoff of SELECTOR has been the development of a memo susbsystem to handle the processing of memos throughout the company. SELECTOR provides a common link between users by prompting the operator when he or she has been sent a memo from some other user. The memo subsystem consists of five separate programs:

- i) a memo entry program that operates in unformatted block mode to accept memos from the user and append them to a transaction file
- ii) an update job that runs constantly in background, waking up every five minutes to process the transaction file
- iii) a character mode memo retrieval program
- iv) a memo print program that is run periodically to print memos for those without terminals, memos that are a day old, and memos that have been flagged for hard copies by senders or receivers
- v) a memo entry subprogram that allows application programs to send memos.

The update job places the fixed-format portion of each memo (sender name(s), receiver name(s), carbon receiver name(s), subject, date, and time) in a memo data base named BASE09. The variable-sized memo text is stored in its own ordinary disc file in a standard group and account (DATA.MEMO). The update job also puts the logon name of each receiver that has a terminal into the MEMOS data set of BASE03 as a signal for SELECTOR to prompt that user to run the memo retrieval program. A personnel data base (BASE06) contains the information necessary to make the correspondence between a user's real name as entered on memos and that user's logon id as returned by the WHO intrinsic.

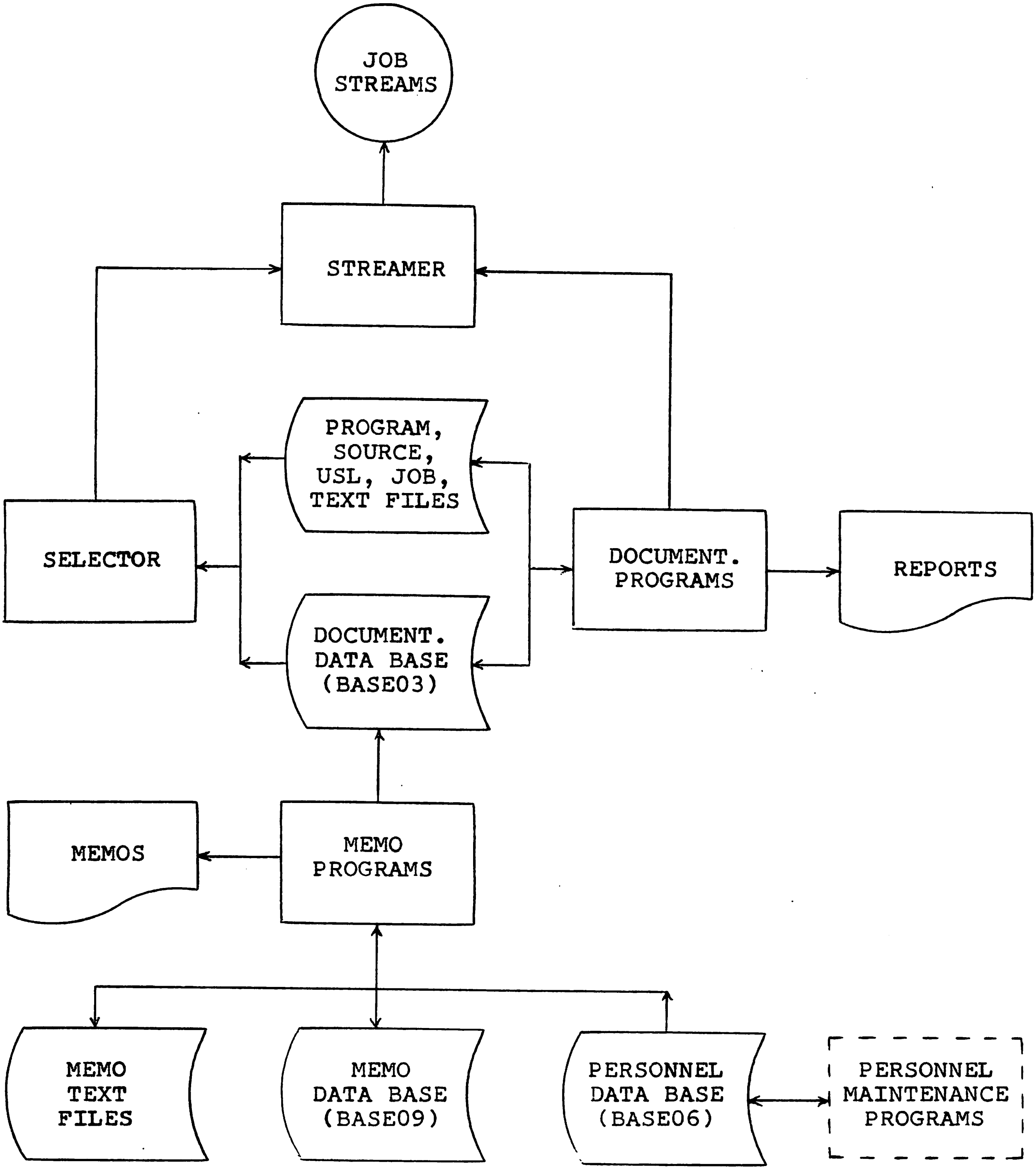
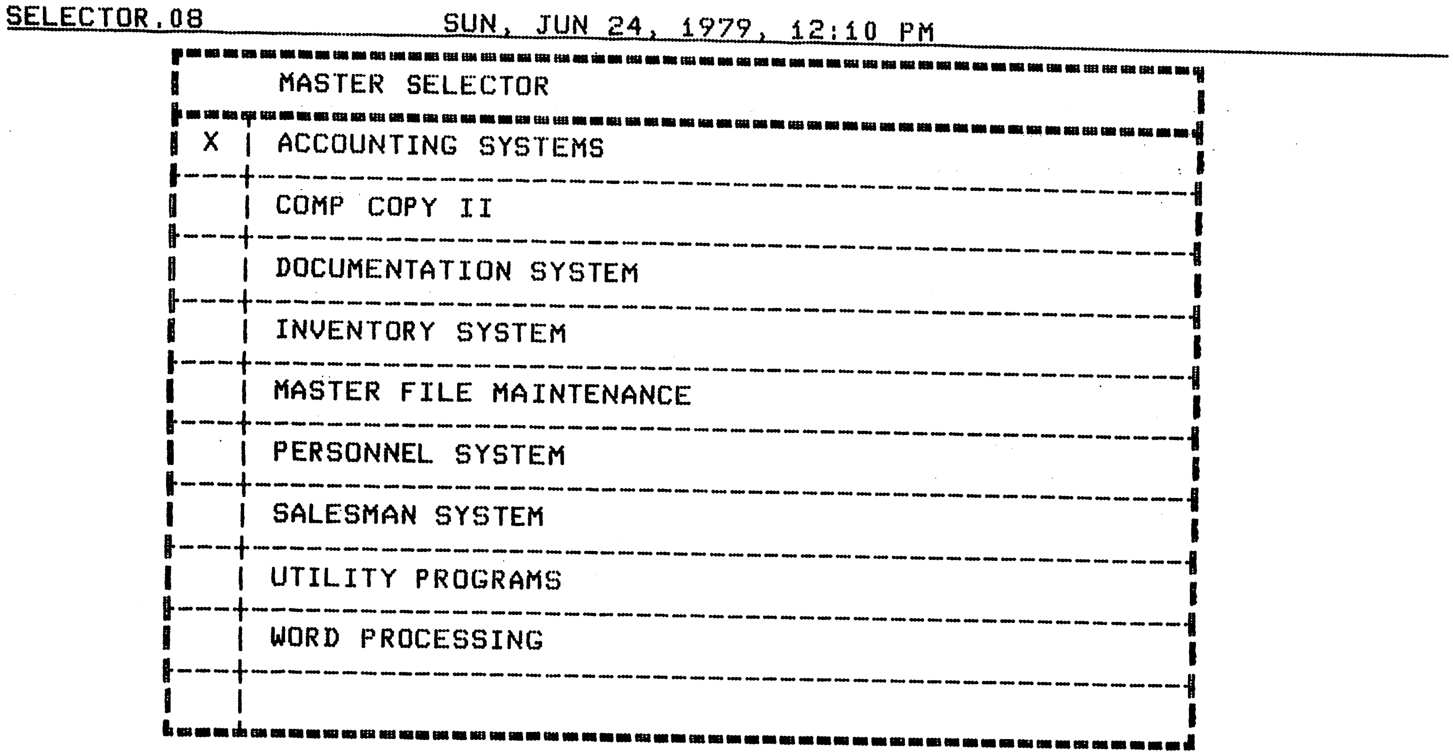

JOB STREAMS
STREAMER
PROGRAM, SOURCE, USL, JOB, TEXT FILES
SELECTOR
DOCUMENT. PROGRAMS
REPORTS
DOCUMENT. DATA BASE (BASE03)
MEMOS
MEMO PROGRAMS
MEMO TEXT FILES
MEMO DATA BASE (BASE09)
PERSONNEL DATA BASE (BASE06)
PERSONNEL MAINTENANCE PROGRAMS

FIGURE 1 - GENERALIZED SYSTEM FLOWCHART

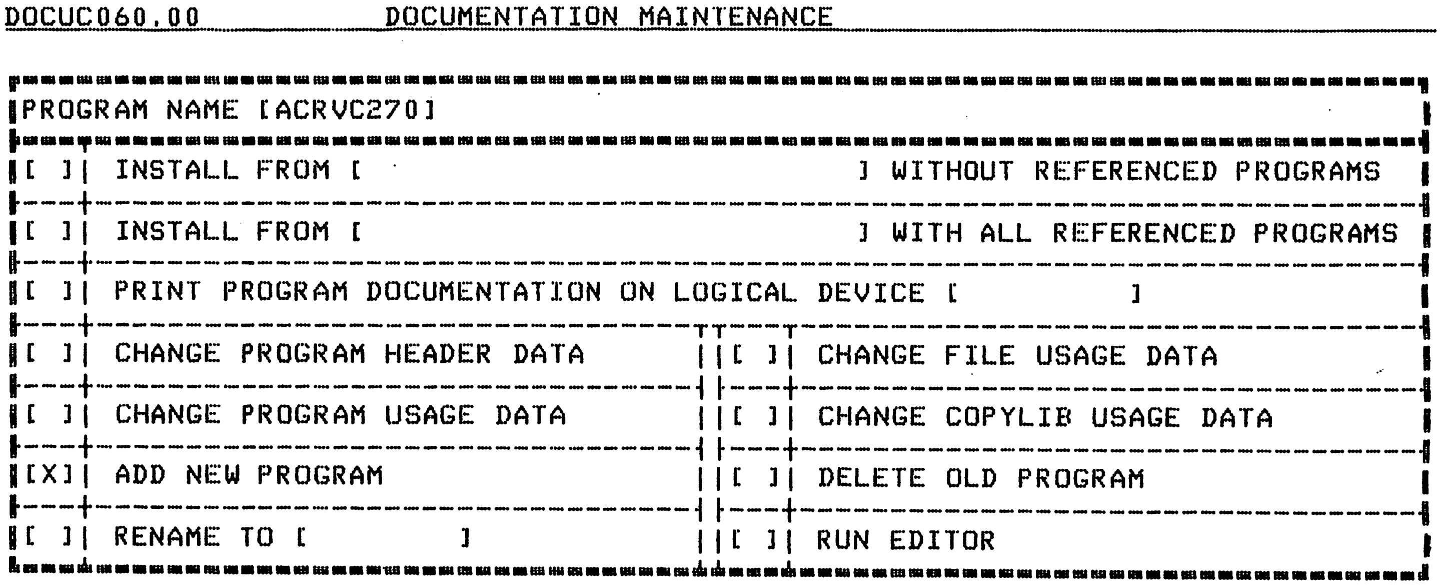
```
NAME:        PROGRAM, DETAIL (/1);
ENTRY:       PROG-ID (PROGRAM-INDEX),
             GROUP-NAME,
             ACCOUNT-NAME,
             PROG-TYPE,
             PROG-DESCRIPTION,
             MNEMONIC-NAME (MNEMONIC-INDEX),
             PROG-CLASS,
             SYSTEM,
             LANGUAGE,
             INSTALLED,
             REVISION,
             NEXT-REV-DUE,
             MAXDATA,
             CAPABILITY,
             MODE,
             ALLOCATE-FLAG,
             STATUS,
             STATUS-DATE;
CAPACITY:    25;

NAME:        DATA-FILE, MANUAL (/1);
ENTRY:       FILE-ID (1),
             GROUP-NAME,
             ACCOUNT-NAME,
             FILE-DESCRIPTION,
             FILE-TYPE;
CAPACITY:    50;

NAME:        COPYLIB, MANUAL (/1);
ENTRY:       COPY-ID (1),
             COPY-DESCRIPTION,
             COPY-TYPE;
CAPACITY:    25;

NAME:        PROG-PROG-XREF, DETAIL (/1);
ENTRY:       CALLING-PROG (PROGRAM-INDEX (CALLED-PROG)),
             CALLED-PROG (PROGRAM-INDEX (CALLING-PROG));
CAPACITY:    50;

NAME:        PROG-FILE-XREF, DETAIL (/1);
ENTRY:       PROG-ID (PROGRAM-INDEX (FILE-ID)),
             FILE-ID (DATA-FILE (PROG-ID)),
             ACCESS-MODE;
CAPACITY:    50;

NAME:        PROG-COPY-XREF, DETAIL (/1);
ENTRY:       PROG-ID (PROGRAM-INDEX (COPY-ID)),
             COPY-ID (COPYLIB (PROG-ID));
CAPACITY:    50;

NAME:        SELECTORS, MANUAL (/1);
ENTRY:       SELECTOR-CODE (1),
             SELECTOR-TITLE;
CAPACITY:    10;

NAME:        OPTIONS, DETAIL (/1);
ENTRY:       SELECTOR-CODE (SELECTORS (OPTION-SEQUENCE)),
             OPTION-SEQUENCE,
             NEXT-SELECTOR,
             PROG-ID;
CAPACITY:    50;
```

FIGURE 2 - PARTIAL SCHEMA OF BASE03



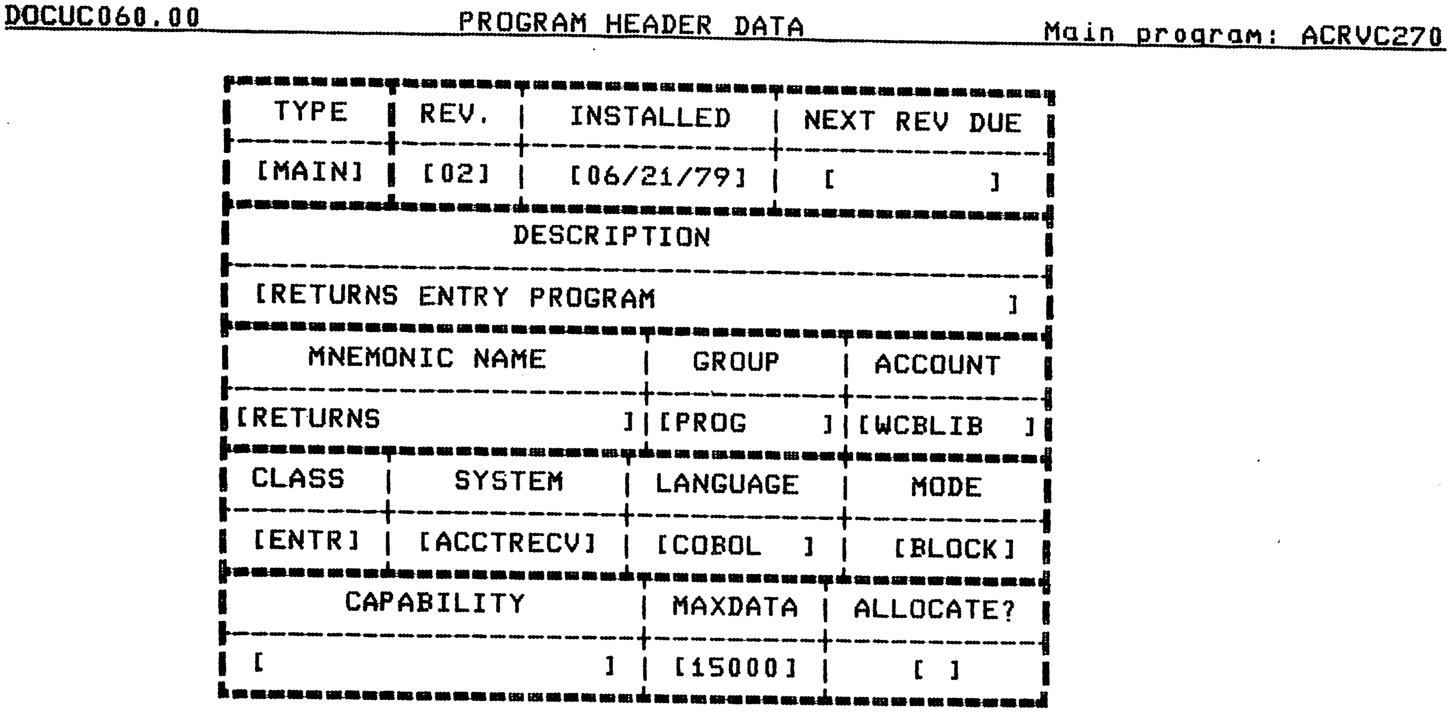
```
SELECTOR.08                  SUN, JUN 24, 1979, 12:10 PM

    MASTER SELECTOR

 X | ACCOUNTING SYSTEMS
   | COMP COPY II
   | DOCUMENTATION SYSTEM
   | INVENTORY SYSTEM
   | MASTER FILE MAINTENANCE
   | PERSONNEL SYSTEM
   | SALESMAN SYSTEM
   | UTILITY PROGRAMS
   | WORD PROCESSING
   |
```

FIGURE 3 - SELECTOR MENU SCREEN



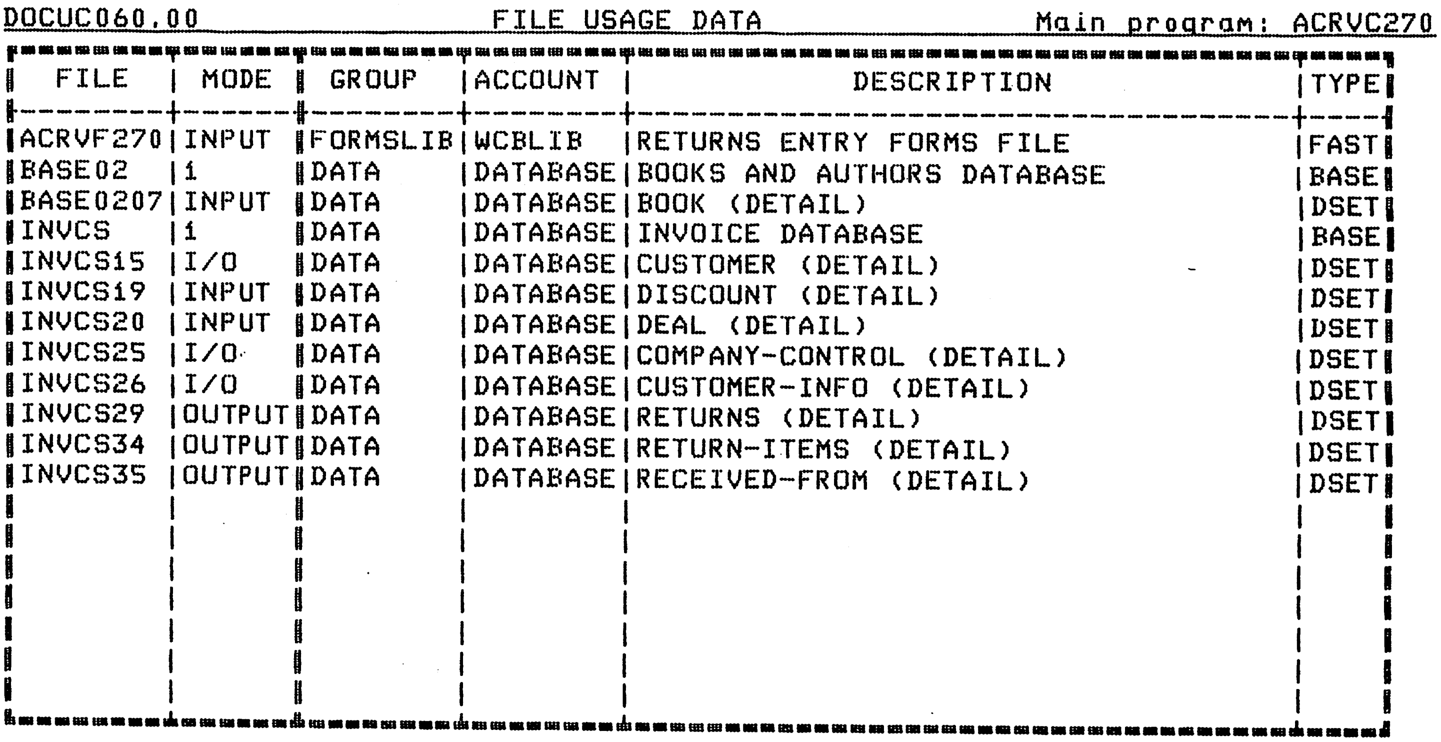
```
DOCUC060.00            DOCUMENTATION MAINTENANCE
+------------------------------------------------------------------------------+
|PROGRAM NAME [ACRVC270]                                                        |
+----+-------------------------------------------------------------------------+
|[ ]| INSTALL FROM [                           ] WITHOUT REFERENCED PROGRAMS    |
+----+-------------------------------------------------------------------------+
|[ ]| INSTALL FROM [                           ] WITH ALL REFERENCED PROGRAMS   |
+----+-------------------------------------------------------------------------+
|[ ]| PRINT PROGRAM DOCUMENTATION ON LOGICAL DEVICE [         ]                 |
+----+-------------------------------------++----+------------------------------+
|[ ]| CHANGE PROGRAM HEADER DATA           ||[ ]| CHANGE FILE USAGE DATA        |
+----+-------------------------------------+|----+------------------------------+
|[ ]| CHANGE PROGRAM USAGE DATA            ||[ ]| CHANGE COPYLIB USAGE DATA     |
+----+-------------------------------------+|----+------------------------------+
|[X]| ADD NEW PROGRAM                      ||[ ]| DELETE OLD PROGRAM            |
+----+-------------------------------------+|----+------------------------------+
|[ ]| RENAME TO [        ]                 ||[ ]| RUN EDITOR                    |
+----+-------------------------------------++----+------------------------------+
```

FIGURE 4 - DOCUC060 MENU SCREEN



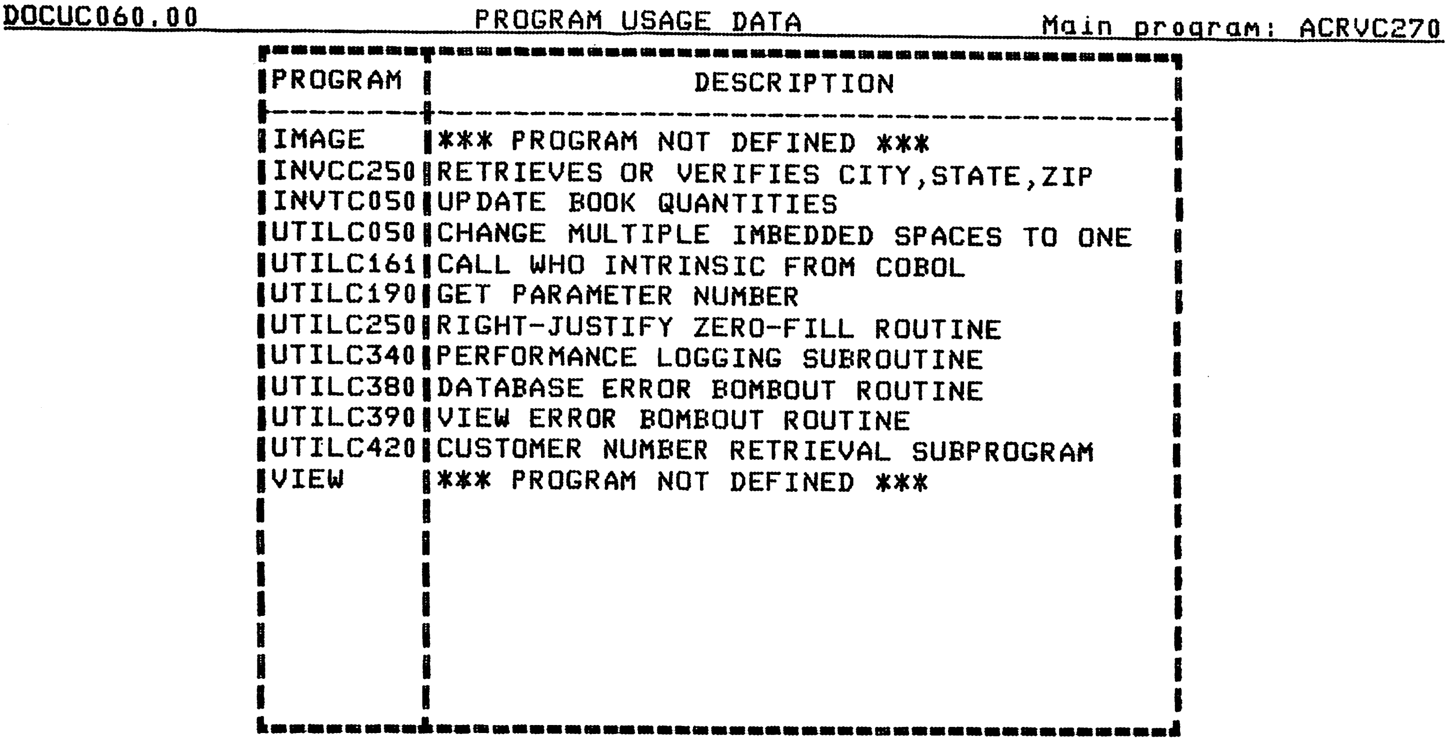
```
DOCUC060.00               PROGRAM HEADER DATA               Main program: ACRVC270

          +-------+------+--------------+-----------------+
          | TYPE  | REV. |  INSTALLED   |  NEXT REV DUE   |
          +-------+------+--------------+-----------------+
          |[MAIN] | [02] |  [06/21/79]  |  [            ] |
          +-------+------+--------------+-----------------+
          |                  DESCRIPTION                  |
          +-----------------------------------------------+
          | [RETURNS ENTRY PROGRAM                      ] |
          +------------------------+----------+-----------+
          |     MNEMONIC NAME      |  GROUP   |  ACCOUNT  |
          +------------------------+----------+-----------+
          |[RETURNS               ]|[PROG    ]|[WCBLIB   ]|
          +-------+----------------+----------+-----------+
          | CLASS |     SYSTEM     | LANGUAGE |   MODE    |
          +-------+----------------+----------+-----------+
          |[ENTR] |   [ACCTRECV]   | [COBOL ] |  [BLOCK]  |
          +-------+----------------+----------+-----------+
          |       CAPABILITY       | MAXDATA  | ALLOCATE? |
          +------------------------+----------+-----------+
          | [                    ] | [15000]  |    [ ]    |
          +------------------------+----------+-----------+
```

FIGURE 5 - DOCUC060 PROGRAM HEADER SCREEN



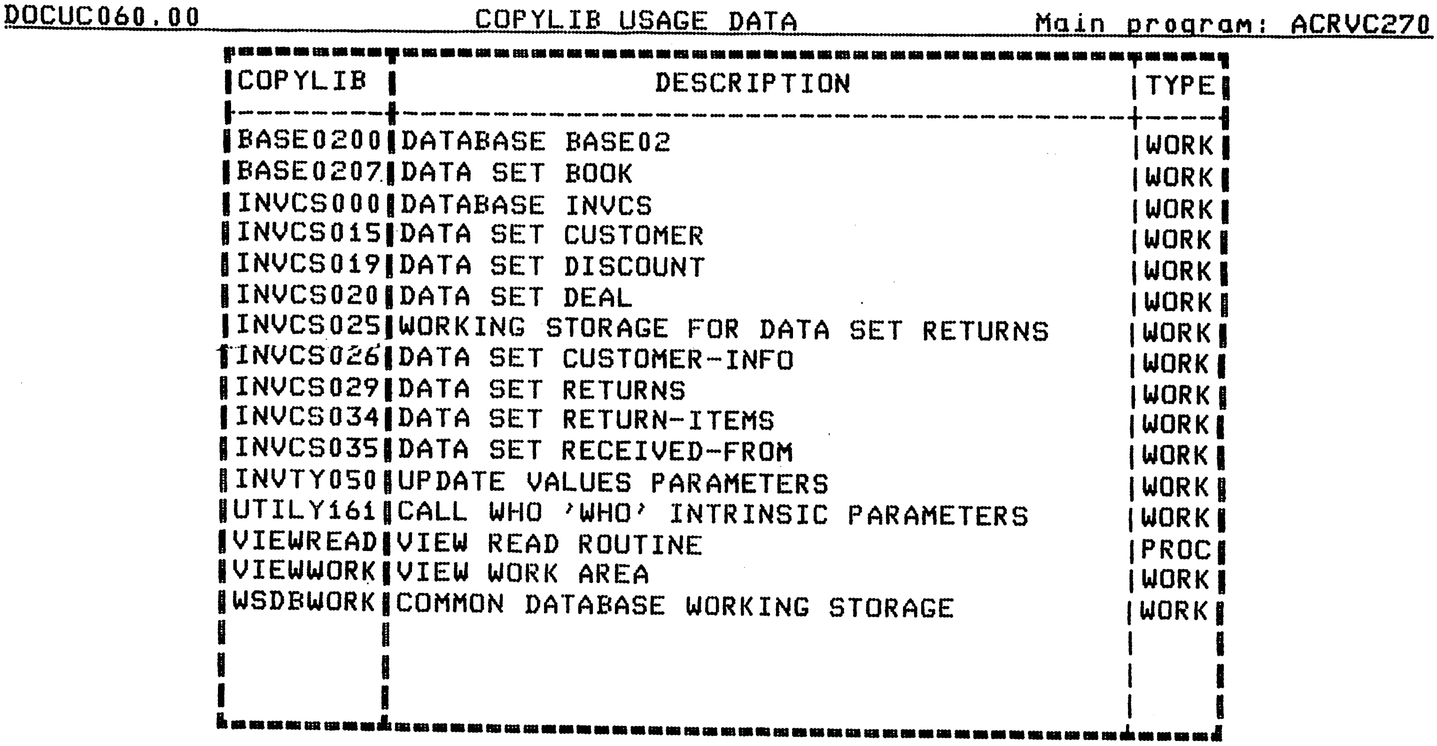
```
DOCUC060.00                  FILE USAGE DATA              Main program: ACRVC270
+---------+-------+---------+--------+-------------------------------------+----+
|  FILE   | MODE  |  GROUP  |ACCOUNT |             DESCRIPTION             |TYPE|
+---------+-------+---------+--------+-------------------------------------+----+
|ACRVF270 |INPUT  |FORMSLIB |WCBLIB  |RETURNS ENTRY FORMS FILE             |FAST|
|BASE02   |1      |DATA     |DATABASE|BOOKS AND AUTHORS DATABASE           |BASE|
|BASE0207 |INPUT  |DATA     |DATABASE|BOOK (DETAIL)                        |DSET|
|INVCS    |1      |DATA     |DATABASE|INVOICE DATABASE                     |BASE|
|INVCS15  |I/O    |DATA     |DATABASE|CUSTOMER (DETAIL)                    |DSET|
|INVCS19  |INPUT  |DATA     |DATABASE|DISCOUNT (DETAIL)                    |DSET|
|INVCS20  |INPUT  |DATA     |DATABASE|DEAL (DETAIL)                        |DSET|
|INVCS25  |I/O    |DATA     |DATABASE|COMPANY-CONTROL (DETAIL)             |DSET|
|INVCS26  |I/O    |DATA     |DATABASE|CUSTOMER-INFO (DETAIL)               |DSET|
|INVCS29  |OUTPUT |DATA     |DATABASE|RETURNS (DETAIL)                     |DSET|
|INVCS34  |OUTPUT |DATA     |DATABASE|RETURN-ITEMS (DETAIL)                |DSET|
|INVCS35  |OUTPUT |DATA     |DATABASE|RECEIVED-FROM (DETAIL)               |DSET|
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
|         |       |         |        |                                     |    |
+---------+-------+---------+--------+-------------------------------------+----+
```

FIGURE 6 - DOCUC060 FILE USAGE SCREEN



```
DOCUC060.00               PROGRAM USAGE DATA              Main program: ACRVC270
                +--------+----------------------------------------+
                |PROGRAM |              DESCRIPTION               |
                +--------+----------------------------------------+
                |IMAGE   |*** PROGRAM NOT DEFINED ***             |
                |INVCC250|RETRIEVES OR VERIFIES CITY,STATE,ZIP    |
                |INVTC050|UPDATE BOOK QUANTITIES                  |
                |UTILC050|CHANGE MULTIPLE IMBEDDED SPACES TO ONE  |
                |UTILC161|CALL WHO INTRINSIC FROM COBOL           |
                |UTILC190|GET PARAMETER NUMBER                    |
                |UTILC250|RIGHT-JUSTIFY ZERO-FILL ROUTINE         |
                |UTILC340|PERFORMANCE LOGGING SUBROUTINE          |
                |UTILC380|DATABASE ERROR BOMBOUT ROUTINE          |
                |UTILC390|VIEW ERROR BOMBOUT ROUTINE              |
                |UTILC420|CUSTOMER NUMBER RETRIEVAL SUBPROGRAM    |
                |VIEW    |*** PROGRAM NOT DEFINED ***             |
                |        |                                        |
                |        |                                        |
                |        |                                        |
                |        |                                        |
                |        |                                        |
                |        |                                        |
                |        |                                        |
                +--------+----------------------------------------+
```

FIGURE 7 - DOCUC060 PROGRAM USAGE SCREEN



```
DOCUC060.00                    COPYLIB USAGE DATA             Main program: ACRVC270
          +--------+------------------------------------------+----+
          |COPYLIB |               DESCRIPTION                |TYPE|
          +--------+------------------------------------------+----+
          |BASE0200|DATABASE BASE02                           |WORK|
          |BASE0207|DATA SET BOOK                             |WORK|
          |INVCS000|DATABASE INVCS                            |WORK|
          |INVCS015|DATA SET CUSTOMER                         |WORK|
          |INVCS019|DATA SET DISCOUNT                         |WORK|
          |INVCS020|DATA SET DEAL                             |WORK|
          |INVCS025|WORKING STORAGE FOR DATA SET RETURNS      |WORK|
          |INVCS026|DATA SET CUSTOMER-INFO                    |WORK|
          |INVCS029|DATA SET RETURNS                          |WORK|
          |INVCS034|DATA SET RETURN-ITEMS                     |WORK|
          |INVCS035|DATA SET RECEIVED-FROM                    |WORK|
          |INVTY050|UPDATE VALUES PARAMETERS                  |WORK|
          |UTILY161|CALL WHO 'WHO' INTRINSIC PARAMETERS       |WORK|
          |VIEWREAD|VIEW READ ROUTINE                         |PROC|
          |VIEWWORK|VIEW WORK AREA                            |WORK|
          |WSDBWORK|COMMON DATABASE WORKING STORAGE           |WORK|
          |        |                                          |    |
          |        |                                          |    |
          |        |                                          |    |
          +--------+------------------------------------------+----+
```

FIGURE 8 - DOCUC060 COPYLIB USAGE SCREEN

```
ACRVC270  RETURNS ENTRY PROGRAM                          07/17/79   ACRVC270    1

TYPE:      MAIN                      CLASS:    ENTR           REV.:  02
NAME:      RETURNS                   SYSTEM:   ACCTRECV       DATE:  06/21/79
GROUP:     PROG                      MODE:     BLOCK          NEXT:    /  /
ACCOUNT:   WCBLIB                    CAP.:
LANGUAGE:  COBOL                     MAXDATA:  15000          ALLOCATE? NO


---------FILES USED-----------------------------------------------------------
NAME      GROUP     ACCOUNT     MODE     DESCRIPTION
------------------------------------------------------------------------------
ACRVF270  FORMSLIB  WCBLIB      INPUT    RETURNS ENTRY FORMS FILE
BASE02    DATA      DATABASE    1        BOOKS AND AUTHORS DATABASE
BASE0207  DATA      DATABASE    INPUT    BOOK (DETAIL)
INVCS     DATA      DATABASE    1        INVOICE DATABASE
INVCS15   DATA      DATABASE    I/O      CUSTOMER (DETAIL)
INVCS19   DATA      DATABASE    INPUT    DISCOUNT (DETAIL)
INVCS20   DATA      DATABASE    INPUT    DEAL (DETAIL)
INVCS25   DATA      DATABASE    I/O      COMPANY-CONTROL (DETAIL)
INVCS26   DATA      DATABASE    I/O      CUSTOMER-INFO (DETAIL)
INVCS29   DATA      DATABASE    OUTPUT   RETURNS (DETAIL)
INVCS34   DATA      DATABASE    OUTPUT   RETURN-ITEMS (DETAIL)
INVCS35   DATA      DATABASE    OUTPUT   RECEIVED-FROM (DETAIL)

--------COPYLIBS USED---------------------------------------------------------
NAME       TYPE       DESCRIPTION
------------------------------------------------------------------------------
BASE0200   WORK       DATABASE BASE02
BASE0207   WORK       DATA SET BOOK
INVCS000   WORK       DATABASE INVCS
INVCS015   WORK       DATA SET CUSTOMER
INVCS019   WORK       DATA SET DISCOUNT
INVCS020   WORK       DATA SET DEAL
INVCS025   WORK       WORKING STORAGE FOR DATA SET RETURNS
INVCS026   WORK       DATA SET CUSTOMER-INFO
INVCS029   WORK       DATA SET RETURNS
INVCS034   WORK       DATA SET RETURN-ITEMS
INVCS035   WORK       DATA SET RECEIVED-FROM
INVTY050   WORK       UPDATE VALUES PARAMETERS
UTILY161   WORK       CALL WHO 'WHO' INTRINSIC PARAMETERS
VIEWREAD   PROC       VIEW READ ROUTINE
VIEWWORK   WORK       VIEW WORK AREA
WSDBWORK   WORK       COMMON DATABASE WORKING STORAGE

-------PROGRAMS CALLED--------------------------------------------------------
NAME       LANGUAGE   DESCRIPTION
------------------------------------------------------------------------------
IMAGE                 PROGRAM NOT DEFINED
INVCC250   COBOL      RETRIEVES OR VERIFIES CITY,STATE,ZIP
INVTC050   COBOL      UPDATE BOOK QUANTITIES
UTILC050   SPL        CHANGE MULTIPLE IMBEDDED SPACES TO ONE
UTILC161   SPL        CALL WHO INTRINSIC FROM COBOL
UTILC190   SPL        GET PARAMETER NUMBER
UTILC250   SPL        RIGHT-JUSTIFY ZERO-FILL ROUTINE
UTILC340   SPL        PERFORMANCE LOGGING SUBROUTINE
UTILC380   SPL        DATABASE ERROR BOMBOUT ROUTINE
UTILC390   SPL        VIEW ERROR BOMBOUT ROUTINE
UTILC420   COBOL      CUSTOMER NUMBER RETRIEVAL SUBPROGRAM
VIEW                  PROGRAM NOT DEFINED
```

FIGURE 9 - SAMPLE PROGRAM DOCUMENTATION

INTERACTIVE PROJECT ANALYSIS AND CONTROL SYSTEM

BY Jeremy Smithers
London Business School

BACKGROUND

Network Analysis is one of the oldest computer applications particularly in the management science field. Over the past twenty five years a large number of packages have been developed for large mainframe computers. These packages were extensively modified to include new facilities as the technique grew more sophisticated. As a result of this project engineers either had to make a considerable investment in understanding how to use these systems or be assisted by a specially trained operator. These systems were also designed for batch processing and although interactive front-ends were added they were mainly simple file editors.

With the advent of large scale time-sharing systems new packages were developed with specific interactive capabilities. However, it has proved difficult to control the cost of project management using these systems. However the same benefits may be achieved by using a suitable package on an in-house mini-computer with the advantage of being able to control the cost.

The major requirements for such a package is that it should be easily usable directly by the project engineers, at the same time providing enough facilities to realistically simulate the project environment.

INTRODUCTION

Interactive Project Analysis and Control System is an on-line package for planning and control of a project written in FORTRAN and designed to operate in a mini-computer time-shared environment.

It is derived from a package called ACSPERT which was developed in 1971 to provide interactive PERT/CPA facilities via a time-sharing bureau. This package has been in operation since 1972 and has been extensively upgraded to meet the requirements of a number of large users. Consequently IPACS derives the benefit of considerable experience of interactive network analysis.

ACSPERT main features are ease of use and efficiency of operation. These features have been retained in IPACS but considerable enhancements have been made the project control area. IPACS provides sufficiently detailed facilities to allow it to control individual resources to the nearest manhour.

IPACS has been developed in a modular open-ended format. This means that new facilities and extra options in existing facilities may be added with the minimum disruption. The file layouts have been designed to record most detailed information but the program is written so that extra information may be easily stored.

There are no absolute limits on the size of network that may be analysed, this being dependant on the amount of memory available. On a 64K byte mini-computer with a time-shared operating system it is designed to handle networks between 3000 - 4000 activities.

FEATURES

- Free format input with default values for all items
- Activities with multiple (complex) resources
- Imposed milestones
- Holidays
- Responsibility codes and costs
- Resource man-hours/week and costs
- Interactive analysis of dates
- Sub-networks
- Hammock activities for overheads
- The Report generator has control facilities
  - to select, group and sort activities
  - to specified special headings and non-standard time periods
- The Report generator produces the following planning reports:-
  - Time Analysis
  - Bar Chart
  - Resource Analysis (tabular/histogram)
  - Cost Analysis
  - Event/Milestone Report

Control Features

- Both the original plan (budget) and the actual implementation may be stored
- The Field Progress report shows the actual and expected progress over the reporting period
- Only exceptions from the forecast progress need be input via the progress commands to up-date the project
- The management cost report shows actual against budgeted progress over the reporting period
- Resource and cost reports show actual and projected actual against budget

SYSTEM OVERVIEW

IPACS provides a system for maintaining a database on a project. The database is contained in two separate files. Firstly the Project Network Details (PND) which contains an entry for each activity. Secondly the Project Reference details which records all other information e.g. titles, milestones, holidays, resources, reponsibility codes, costings, etc.

A new project is initially set up by using the NEW command. This will ask for some reference details and a specification of each activity. After a new project is established HOLIDAYS and MILESTONES may be entered and RESOURCE and RESPONSIBILITY CODE profiles defined. These commands also provide facilities for editing and listing all these details.

The network data may be subsequently EDITED where there are a set of facilities for adding/overwriting activities, modifying individual fields and listing selected activities based on criteria about individual fields. Activities may also be entered as SUB-NETWORKS from a library. This means that standard tasks need not be redefined each time they occur.

A project may be ANALYSED to establish the earliest and latest dates for each activity taking into consideration imposed milestones and holidays. All activities are considered to start at this earliest date unless they are SLIPPED back.

Once a time schedule has been established for a project then a series of reports may be produced. These show the time schedule of the project in clear tabular form (TIME ANALYSIS) or in graphical form (BAR CHART). The resource requirements (RESANAL) and costs (COSTANAL) can be shown in tabular or histogram form. An EVENT (or Milestone) report shows key dates. Each report may be controlled as to which activities should be analysed, the sequence and grouping of the activities, the dates covered, and special formatting requirements.

The initial analysis of the project is the planning and evaluation stage. Once a plan for the project has been finalised then the updating mode is changed to ACTUAL. This preserves the initial plan as a budget against which to measure the actual progress of the project.

Updating and monitoring the project is done on a reporting cycle. Firstly a Field Progress Report is produced showing for all current activities the actual and forecast progress up to the reporting date. This report is used on an exception basis, only differences from the forecast need be recorded by the field supervisors. These differences are entered via the PROGRESS command which automatically progresses all other activities in line with the forecast. All the same time as the project is progressed, a Management Cost Report is produced showing the actual progress compared with the budget. At this time the next FPR can be produced which starts the next loop of the reporting cycle.

A FRENCH ACCOUNTING PACKAGE : GAEL 3

AN EXERCICE IN PORTABILITY FROM IBM TO HP 3000

by CLAUDE NOVARESIO, Product Manager, SLIGOS CPP
and THIERRY VIDELAINE, Former EDP Manager, PLASTIC OMNIUM

A. THE PACKAGE

1. General presentation

SLIGOS - set up GAEL 3 on the bases of knowledge gained from the experience of using a product already existant - that is, a general accounting program used in bureau service for several years.

GAEL 3 has the following aims :

- to enable an easy adaptation to the management methods of each company thanks to a highly developed personnalization. Input document, data composition, chart of accounts, are customized to the needs of each company.
- to facilitate a rational processing of the general, analytical and budgetary accounting.
- to be able to handle several independant accounting procedures simultaneously.
- to establish an accounting data base containing the data required for specific additionnal applications.
- to be used very easily.

2. Who is concerned by GAEL 3

a. data processing departments

GAEL 3 offers a processing philosophy which greatly simplifies processing handlings.

the technical point of view

- a single point of entry is used to receive each user's data whether this concerns the updating of permanent files or accounting entries.
- no set format for receiving the data is stipulated, thus each user may design his own entry forms and this makes it possible to integrate GAEL 3 into an already existing data processing system by simply providing for the necessary interfaces.
- the printing of the results is standardized, and a single accounting document is used for the printing.
- the processing progression is automatic according to user's request.
- for large numbers of companies the system adopts a processing by lots of a thousand users maximum, taking over the control of the file belonging to each lot.
- service routines enable the general back up of the data base, its reorganisation, printing or interrogation.
- program controls protect the files during their processing and their updating.

b. accounting departments

- in the presentation of results GAEL 3's classical form allows it to be adopted by all.
- in general accounting all the accounts of clients and suppliers can be integrated into the chart of accounts, though remaining independant to be used and printed separately.
- the data acquisition, being in variable format, makes it possible for each user to :
  - employ his personnalized input documents
  - introduce the useful data without any restriction while,

however, also treating specific information, if desired, via interfaces if needed.

- the system allows a firm to handle several accounting procedures for its subsidiaries, agencies, factories, etc...

- in addition to a general accounting system and a budgetary control GAEL 3 offers two types of analytical accounting which may remain fully independant (by organizational unit, project, fund, activity a.s.o.)

3. to both of them

- each user may have his own chart of accounts and basic document when he has several accounting systems.

- during the same processing run the system independently handles the bookkeeping problem of each user : current period, end of period, end of year, a.s.o.

- the system can automaticaly correct certain errors (in totalling, coding...) in order not to delay the processing of bookkeeping.

- the chart of accounts can be adapted to different fiscal laws, and the system can handle entries presented in terms of foreign currencies.

- upon request the system provides enough data to enable an accurate invoicing of the users when an internal invoicing procedure exists in the firm.

4. Caracteristics

GAEL 3 is written in ANS Cobol, and it is thus adaptable to most of the systems presently on the market.

A minimum configuration of two disk units is required. Tape units are optional however during certain phases of processing their use is strongly recommended.

The ideal configuration is three disks units and two tapes units.

SLIGOS has developped an original technique for the organization or files to set up the data base ; the performances obtained are not expensive, whatever the volume processed.

GAEL 3 is available for the following computers.

- IBM 3
- IBM 360/370 DOS and OS
- CII HB 64 series
- BURROUGHS small and medium
- HEWLETT PACKARD 3000

Caracteristics of the data base :

length of account record	:	300 bytes
length of bookkeeping entry	:	134 bytes

B. ADAPTING GAEL 3 FROM IBM/370 TO HP/3000 COBOL

Within 2 weeks of the order, SLIGOS supplied a mag tape comprising 16 source programs,and about 25 000 lines of COBOL Code, as well as a listing of the tape contents, and a complete set of manuals.

The translation process included four steps.

1. Starting from the original tape (encoded in EBCDIC), translation into ASCII, creating and formatting the GAEL source library, extracting and storing into different source files the various programs. This was done using FCOPY.

2. Checking the source library to delete unnecessary members, and adapting the others to the HP/3000 COBOL format. This was done using EDITOR. At this point, an EDITOR USE file was developped for step 3.

   There were two main reasons for source members to be deleted from the library :

- discrepancies between the formats of the ENVIRONMENT DIVISION ASSIGN system name in HP/3000 and IBM/370 COBOL.

- wholesale deletion of anything related to the IBM ISFMS access method, to be replaced by CALL's to KSAM intrinsics.

3. Using EDITOR, conversion of source programs from IBM/370 to HP/3000 COBOL, computing, preparing and saving into a newly-created account.

   the main problems encountered on this phase were:

   a. converting ISFMS i/o statements to KSAM CALL's. This was done using specially designed source library members, and extensive use of the COPY... REPLACING... BY... clause.

   b. Rewriting GVLMVT, a direct access i/o module, into SPL (HP/3000 COBOL does not currently support a REWRITE clause)

   GVLMVT was the sole module to be rewritten.

   c. Incorporating segmentation into programs. This process was made much easier by the fact that they were written in a truly professional and consistent manner.

   d. Writing form scratch the various batch jobs, using JCW'S to simulate OS/370's conditional JCL.

4. Testing twice :

   - with test data supplied by SLIGOS, and checking with results produced on an IBM/370 computer.

   - processing a month's actual data, and checking with the results produced by the current accounting program.

The whole translation process took about one man's month.

OVERCOMING OPERATIONAL MANAGEMENT PROBLEMS WITH A SCHEDULER

****

by

Mr C H Oldfield
The London Business School

INTRODUCTION

In September 1977 the London Business School Computer Department installed a HP 3000 Series II Model 8 Computer System to provide replacement computing facilities to the research and administrative departments, who were previously dependent upon an ageing IBM 360/65, at University College of London, for providing their computing needs. The HP 3000 was to supply interactive and batch computer facilities to users in the following research areas: Economic Forecasting; Finance; Accounting; Marketing; Statistics; Quantitative Methods; Modelling. The system was also to provide a service to a mix of administrative, teaching and external users.

Initially, the first six month period was virtually devoted to the conversion of existing software from the IBM 360/65 with a substantial amount of new software being developed.

It was after this period and a great deal of system experience had been gained both by users and operations personnel, that it became apparent that the system lacked some desirable operational control functions in the following areas:

- Enforcing job CPU resource limitations
- Scheduling and providing notification of tape requests
- Cancellation of jobs by users prior to execution.

After careful deliberations in attempting to find a solution to overcome these system deficiencies it was decided that by implementing some form of automatic monitoring and scheduling system that this would provide the means of surmounting these problems.

OPERATIONAL MANAGEMENT PROBLEMS

In an effort to maximise the general batch job throughput and to evenly distribute batch CPU resources between users, during peak usage periods, it was decided that a realistic maximum CPU time obtainable would have to be placed on jobs during these periods. Unfortunately, we found that the system lacked any useful mechanism which could be adopted to enforce certain job CPU resource limitations within different time slots. (e.g. Setting a maximum CPU time of 300 seconds during Prime Shift). It is true that by using variations of different input priorities some form of control could be gained. This option though, was not totally secure as some measure of trust has to be placed upon the users to submit work with the correct parameters. A side effect of introducing a batch CPU resource limitation, would be to encourage users to segment work into smaller packages in order to obtain the minimum turnaround time.

Due to our moderately active tape environment we found that our operations personnel needed some form of advance notification system of tape requirements prior to access. This would then enable tapes to be obtained from the tape library and scheduled for loading with the correct security status (read/write permit) and in the correct order. Providing the system to be adopted catered for obtaining tape details, up to the point of job termination, it would eliminate the confusion when handling volume tape sets that comprise of more than one tape.

As most HP 3000 system managers/operators will be aware that the console command =RECALL will not operate once the end of tape condition has been reached on the first tape, leaving the operator to guess what the next tape required is. Our experiences also showed that a great deal of errors and time were incurred in a free-for-all tape request/Mounting environment as requests frequently conflicted with available devices. In an attempt to eliminate these problems we felt that it would be necessary to prohibit all tape accesses from sessions. Initially it appears to place a great hardship upon the user but our experiences have shown that it is sometimes preferable to place tape work in the batch queue as in a heavily loaded system the session elapsed period can be very lengthy.

Lastly, one other drawback we found was that a user could not cancel a job once it had been streamed. Again, in our experience we found users often realise some alteration to the job file is necessary immediately after submission which meant the users having to interrupt the operators to effect the cancellation.

DESIGN PHILOSOPHY

Once agreement had been reached that a monitoring and scheduling system would provide the optimum solution in overcoming these problems, it was necessary to generate the general design philosophy to be adopted. Primarily and foremost the total package would have to be easy to use for both experienced and inexperienced users. It was therefore essential that the package catered for both these types of users by incorporating a dual mode operation with meaningful error messages and a general 'Help' facility. The package was to consist of common routines, for both users and operations personnel, utilising only standard features supported within MPE (i.e. Non-Privileged Mode). Since total monitoring and scheduling would be required it would be necessary to probibit users from using the STREAM command. Therefore identical facilities to that of the STREAM command would have to be supported within the package, including syntax checking on job statements. Jobs submitted via the scheduler would be vetted to ensure that the CPU resource limitations applicable were enforced, deferring any jobs exceeding the limitations. The scheduler would provide the means to schedule work on an immediate, specific time and date or every day basis. Users submitting tape work would be required to include 'COMMENT SETUP' statements, within their job-files with the appropriate tape details coded in the remaining portion of the statement. These statements would be copied to an operator inspection file on submission of the job. Alternatively, these statements could be used to relay other pertinent information to the operator. Jobs submitted with embedded 'COMMENT SETUP' statements would automatically be placed into a hold status, pending release by operations personnel. Users would be allowed to cancel owned jobs or modify scheduling parameters on their jobs through the scheduler package. The same functions, would be available to the operations personnel, but with the ability to override scheduling limitations and ownership basis.

USER REACTION

On announcing to the users the proposed implementation of a scheduling system, the immediate feedback was that of scepticism and objection to change. To counteract this it was decided to introduce the 'Scheduler' in unison with the standard STREAM facility for a trial period and obtain feedback from users during and after this period. The package became available for use in December 1978. The initial transfer of users to the scheduler was at first very low but once reluctant users realised the benefits offered by the package a steady growth in numbers was seen. So much so, that it was decided to make the use of the 'Scheduler' mandatory from February 1979. Since the introduction of MIT 1906 software release we have replaced the STREAM command with a system wide user defined command which invokes the scheduler thereby preventing any user escaping the scheduling limitations.

User reaction and comments has reversed to that of being favourable since the initial design goals have been achieved and that the package provides flexibility over the previous operation.

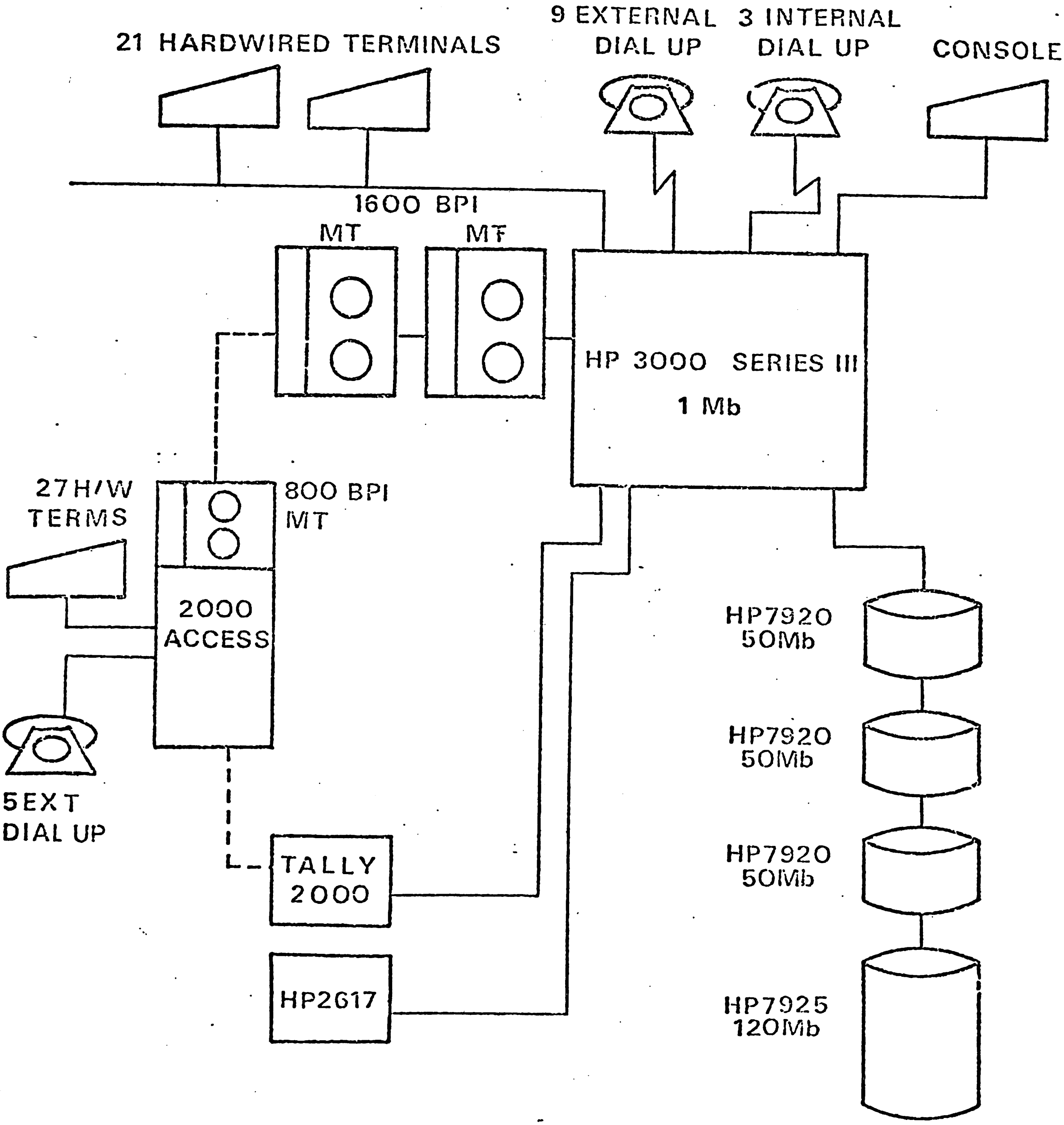
FURTHER BENEFITS

Additional benefits have been gained since the introduction of the 'Scheduler' package in that a large 'WAIT' queue is not built up which is subject to loss when the system cannot be successfully 'WARMSTARTED' after a system failure. A relatively new feature added to the 'Scheduler' package is in handling and scheduling of jobs to be transmitted via MRJE/3000 to a host-system. This has enabled us to provide the user with a common interface whether work is to be scheduled on the HP 3000 or via MRJE/3000. The user is not required to become familiar with the MRJE/3000 subsystem features and commands.

PROPOSED ENHANCEMENTS

Currently we are considering whether to implement further features in the form of job correlation and true resource scheduling. Although the present system provides a list of the previous twenty-four hours job statements and an hourly status report on input, output and execution queues, we feel the probable need for complete job correlation from the input stage to the final output stage. The true resource scheduling feature may be a requirement of the future where jobs are not scheduled according to a start time but scheduled when resources become available within various limitations. To date though no final conclusions have been reached whether these facilities will be implemented.

LBS COMPUTER CENTRE
HARDWARE CONFIGURATION


21 HARDWIRED TERMINALS
9 EXTERNAL DIAL UP
3 INTERNAL DIAL UP
CONSOLE
1600 BPI
MT
MT
HP 3000 SERIES III
1 Mb
27 H/W TERMS
800 BPI MT
2000 ACCESS
5 EXT DIAL UP
TALLY 2000
HP2617
HP7920 50Mb
HP7920 50Mb
HP7920 50Mb
HP7925 120Mb

SYSTEM OVERVIEW

USERS:

- 25/30 Research/Teaching Faculty
- 40 Doctoral Students
- 10 "Computer-Related" personnel
- 25/30 External Users

APPLICATIONS:

Current:

- Economic Forecasting
- Finance
- Marketing
- Accounting
- Statistics
- Quantitative Methods
- Modelling

Future:

- Expansion of several of the above in particular:
- Economic Forecasting
- Financial Services

SYSTEM OVERVIEW
Continued

"TYPICAL" DAILY WORK-LOAD:

- Production Execution
- Fortran Program Development
- Basic Program Development

DURING "PRIME-SHIFT" A SYSTEM MIX OF:

- 10/15 Sessions
- 2 Executing Batch Jobs
- 1 Executing System Monitor

COMPILER USAGE:

- Fortran	80%
- Basic	19%
- SPL	1%

OPERATIONAL MANAGEMENT PROBLEMS

LACK OF OPERATIONAL CONTROL FUNCTIONS IN THE FOLLOWING AREAS:

- Enforcing Job CPU Resource Limitations
- Scheduling and providing notification of tape requests
- Cancellation of jobs by users prior to execution

ENFORCING JOB CPU RESOURCE LIMITATIONS TO:

- Maximise general job throughput during peak usage periods
- Evenly distribute batch CPU resources during peak usage periods

CONTROL BY:

- Prohibit users from using the STREAM command
- Imposing a maximum CPU time obtainable during peak usage period
- Deferring jobs exceeding limitations

SCHEDULING AND PROVIDING NOTIFICATION OF TAPE REQUESTS TO:

- Provide operations personnel with advance notification of tape requirements prior to access
- Enable correct security status (Read/Write) to be applied to tape volumes prior to loading
- Eliminate the confusion arising when handling volume tape sets that comprise of more than one tape
- Reduce number of errors incurred through incorrect loading
- Reduce number of request conflicts for available devices

CONTROL BY:

- Prohibit all tape accesses from sessions
- Tape requesting jobs required to contain 'COMMENT SETUP' statements with appropriate tape details coded

CANCELLATION OF JOBS BY USERS PRIOR TO EXECUTION TO:

- Relieve interruptions on operations personnel by users

SCHEDULER DESIGN PHILOSOPHY

- Primary and foremost easy to use for both experienced and inexperienced users
- Dual mode operation for both these types of users
- Meaningful error messages
- General 'HELP' facility
- Common routines for both users and operations personnel
- Utilise only standard features supported within MPE (Non-Privilege mode)
- Provide identical facilities to that of the STREAM command including syntax checking on job statements
- Scheduler to impose resource limitations applicable, deferring any jobs exceeding the limitations
- Users to be able to schedule work on an immediate, specific time and date, or every day basis
- Jobs submitted with 'COMMENT SETUP' statements to be placed into a 'HOLD' status, pending release by operations personnel
- Users to be able to cancel owned jobs
- Users to be able to modify scheduling parameters on owned jobs
- Same functions available to operations personnel but ability to overide scheduling limitations and ownership

USER REACTION

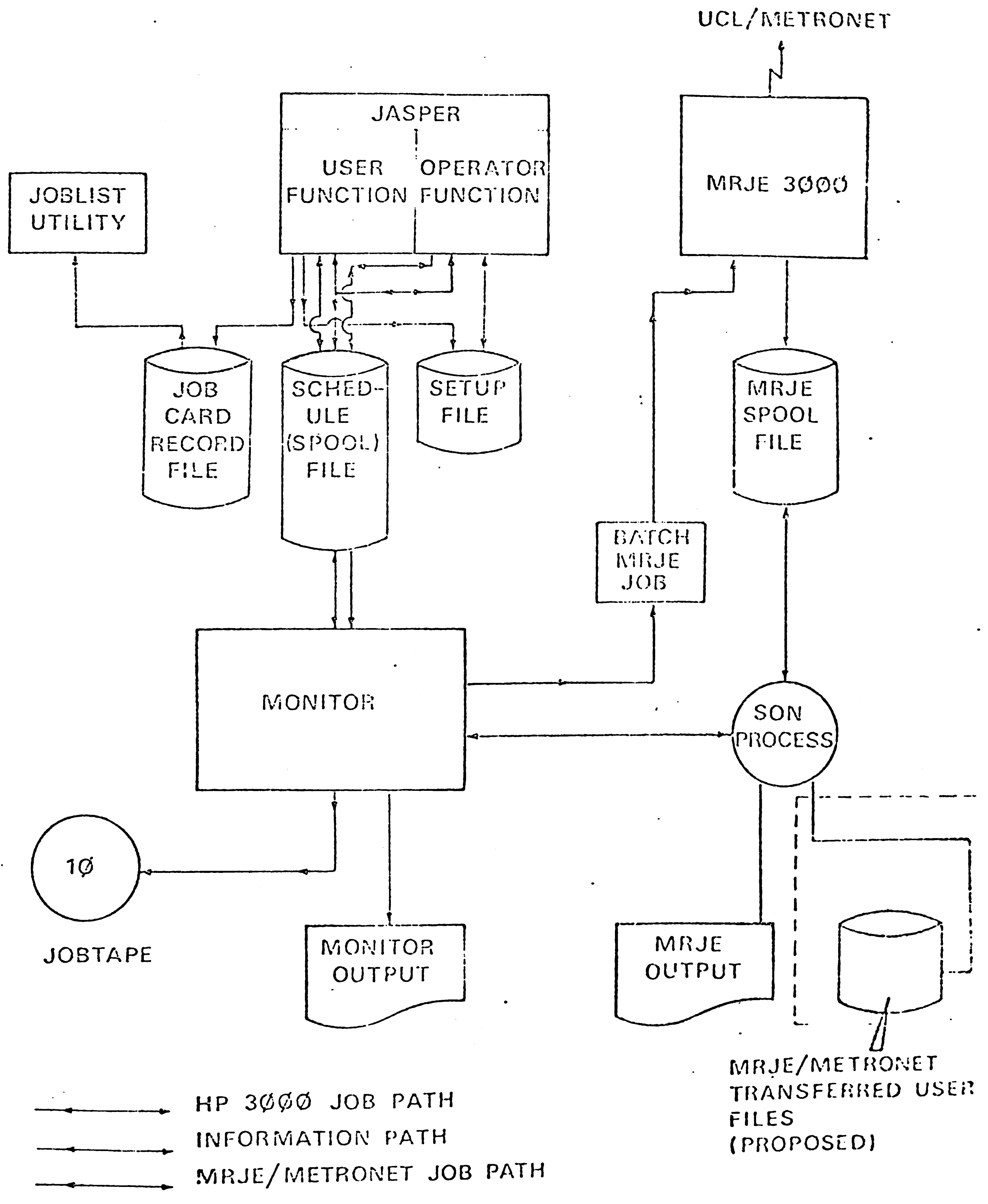
ON PROPOSAL OF IMPLEMENTING A SCHEDULING SYSTEM:

- Initial feedback, objection to change and scepticism
- To counteract this, Scheduler to be introduced in unison with standard 'STREAM' facility. December 1978
- Transfer of users to scheduler initially low
- Once reluctant users realised scheduler benefits, steady growth in numbers was seen
- Scheduler use made mandatory from February 1979
- Since introduction of MIT 1906 'STREAM' command replaced with UDC to invoke the scheduler
- User comments favourable of scheduler package

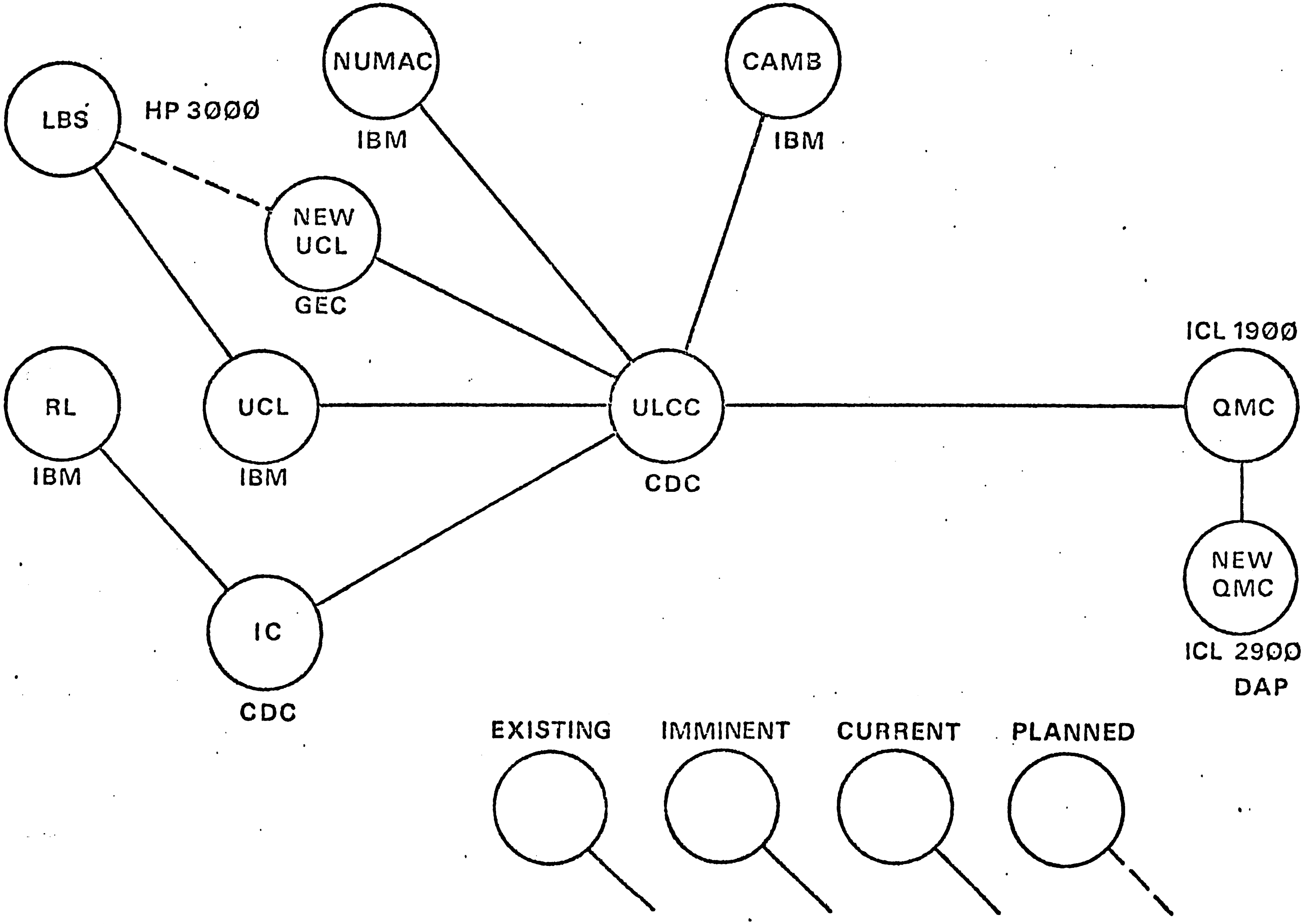
FURTHER BENEFITS

- Large 'WAIT' queues are not built up, which are subject to loss after a system failure
- New feature, where work can be submitted to HP 3000 or via MRJE/3000 to host system
- Common interface, users not required to be familiar with MRJE/3000 subsystem

LBS SCHEDULING SYSTEM


UCL/METRONET
JASPER
USER FUNCTION
OPERATOR FUNCTION
MRJE 3000
JOBLIST UTILITY
JOB CARD RECORD FILE
SCHEDULE (SPOOL) FILE
SETUP FILE
MRJE SPOOL FILE
BATCH MRJE JOB
MONITOR
SON PROCESS
10
JOBTAPE
MONITOR OUTPUT
MRJE OUTPUT
MRJE/METRONET TRANSFERRED USER FILES (PROPOSED)
HP 3000 JOB PATH
INFORMATION PATH
MRJE/METRONET JOB PATH

LONDON UNIVERSITY METRONET NETWORK


NUMAC
IBM
CAMB
IBM
LBS
HP 3000
NEW UCL
GEC
RL
IBM
UCL
IBM
ULCC
CDC
ICL 1900
QMC
NEW QMC
ICL 2900
DAP
IC
CDC
EXISTING
IMMINENT
CURRENT
PLANNED

```
:STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979,  2:48 PM
COMMAND? HELP

These are the commands available under JASPER:
The parameters in {} are mandatory,those in [] may be defaulted.
ADD (A)    {filename}[group][account][run-date][run-time]
MODIFY (M) {filename}[group][account][run-date][run-time]
LIST (L)
DELETE (D)
SHORT (S)  {command}{filename}[group][account][date][time]
METRONET (MET)
BUILD (B)
HELP (H)
EXIT (E)
Which would you like a description of? 'EXIT' will return you to
the sheduler.
ENTER COMMAND  >    ADD

ADD                  Adds a previously built job file to the spool file.
                      You will be prompted for parameters filename,group,
                      account,desired run-date,and run-time.
ENTER COMMAND  >    LIST

LIST                 Enables selective listing of jobs in the spool file:
         L      - provides a list of your HP3000 jobs
         L@     - provides a list of all users' HP3000 jobs.
         LM     - provides a list of your METRONET jobs.
         L@M    - provides a list of all users' METRONET jobs.
ENTER COMMAND  >    EXIT

COMMAND? E

END OF PROGRAM
:
```

```
STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979, 11:53 AM
COMMAND? ADD

NAME OF FILE? MCLOGS

GROUP?HARRIS

ACCOUNT?COMPUTER

MCLOGS.HARRIS.COMPUTER
IS THIS THE CORRECT JOB NAME (Y OR N) ? Y

DATE OF RUN?(DD/MM/YY)
ENTER HOUR OF DAY JOB IS TO BE RUN
BETWEEN 1 TO 24  OR ENTER 00 IF JOB CAN BE RUN AT ANY      TIME

RUN DATE= 16/07/79  RUN TIME=      0  HRS
IS THIS CORRECT? Y

PRIME SHIFT RUN PROHIBITED
RESCHEDULED FOR22 HRS
MCLOGS.HARRIS.COMPUTER      SUCCESSFULLY SCHEDULED
COMMAND? L

                    LBS SCHEDULER                PAGE     1
  JOB        GROUP         ACCOUNT   RUN DATE     TIME     METRONET
COMMAND? L@


                    LBS SCHEDULER                PAGE     1
  JOB         GROUP        ACCOUNT   RUN DATE     TIME     METRONET
 NIGHTJOB    LOGFILES    SYS         EVERY DAY   11:00PM
 REPORT20    PUB         LEASE       EVERY DAY    6:00PM
 ROYALCJ     LOGFILES    SYS         EVERY DAY   12:00AM
 MCLOGS      HARRIS      COMPUTER    16/07/79    10:00PM
COMMAND? E

END OF PROGRAM
:
```

```
:STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979, 12:01 PM
COMMAND? A

NAME OF FILE? MCLOGS

GROUP?

ACCOUNT?

MCLOGS.HARRIS.COMPUTER
IS THIS THE CORRECT JOB NAME (Y OR N) ? Y

DATE OF RUN?(DD/MM/YY)
ENTER HOUR OF DAY JOB IS TO BE RUN
BETWEEN 1 TO 24  OR ENTER 00 IF JOB CAN BE RUN AT ANY      TIME

RUN DATE= 16/07/79  RUN TIME=      0  HRS
IS THIS CORRECT? Y

PRIME SHIFT RUN PROHIBITED
RESCHEDULED FOR22 HRS
MCLOGS.HARRIS.COMPUTER      SUCCESSFULLY SCHEDULED
COMMAND? L

                      LBS SCHEDULER                  PAGE    1
  JOB        GROUP       ACCOUNT   RUN DATE        TIME     METRONET
 MCLOGS     HARRIS      COMPUTER   16/07/79      10:00PM
COMMAND? L@

                      LBS SCHEDULER                  PAGE    1
  JOB        GROUP       ACCOUNT   RUN DATE        TIME     METRONET
 NIGHTJOB   LOGFILES    SYS        EVERY DAY     11:00PM
 REPORT20   PUB         LEASE      EVERY DAY      6:00PM
 ROYALCJ    LOGFILES    SYS        EVERY DAY     12:00AM
 MCLOGS     HARRIS      COMPUTER   16/07/79      10:00PM
COMMAND? LM

                      LBS SCHEDULER                  PAGE    1
  JOB        GROUP       ACCOUNT   RUN DATE        TIME     METRONET
COMMAND? L@M

                      LBS SCHEDULER                  PAGE    1
  JOB        GROUP       ACCOUNT   RUN DATE        TIME     METRONET
 DMP04      PUB         OPERATOR   13/07/79      *IMMED*      SENT
 Q1941050   REQ         OPERATOR   13/07/79      *IMMED*      SENT
 DMP08      PUB         OPERATOR   13/07/79      *IMMED*      SENT
COMMAND? D

NAME OF FILE? MCLOGS

GROUP?

ACCOUNT?

MCLOGS.HARRIS.COMPUTER
IS THIS THE CORRECT JOB NAME (Y OR N) ? Y
MCLOGS.HARRIS.COMPUTER      DELETED
COMMAND? E

END OF PROGRAM
:
```

```
:STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979, 12:09 PM
COMMAND? L

                    LBS SCHEDULER                      PAGE    1
  JOB         GROUP        ACCOUNT   RUN DATE       TIME    METRONET
 Q1971204    HARRIS      COMPUTER   16/07/79     SETUP=1
COMMAND? S
   KEY PARAMETERS PLEASE    D;Q1971204

Q1971204.HARRIS.COMPUTER    DELETED
COMMAND? S
   KEY PARAMETERS PLEASE    A;MCLOGS

MCLOGS.HARRIS.COMPUTER
RUN DATE= 16/07/79  RUN TIME=      0  HRS
PRIME SHIFT RUN PROHIBITED
RESCHEDULED FOR22 HRS
MCLOGS.HARRIS.COMPUTER      SUCCESSFULLY SCHEDULED
COMMAND? S
   KEY PARAMETERS PLEASE    D;MCLOGS

MCLOGS.HARRIS.COMPUTER      DELETED
COMMAND? S
   KEY PARAMETERS PLEASE    A;MCLOGS;99/99/99;02

MCLOGS.HARRIS.COMPUTER
RUN DATE= 99/99/99  RUN TIME=      2  HRS
JOB SCHEDULED FOR  2HRS
MCLOGS.HARRIS.COMPUTER      SUCCESSFULLY SCHEDULED
COMMAND? L

                    LBS SCHEDULER                      PAGE    1
  JOB         GROUP        ACCOUNT   RUN DATE       TIME    METRONET
 MCLOGS      HARRIS      COMPUTER   EVERY DAY     2:00AM
COMMAND? S
   KEY PARAMETERS PLEASE    M;MCLOGS.HARRIS.COMPUTER;17/07/79;03

MCLOGS.HARRIS.COMPUTER
RUN DATE= 17/07/79  RUN TIME=      3  HRS
   JOB SCHEDULED FOR  3 HOURS
   JOB MODIFIED
COMMAND? L

                    LBS SCHEDULER                      PAGE    1
  JOB         GROUP        ACCOUNT   RUN DATE       TIME    METRONET
 MCLOGS      HARRIS      COMPUTER   17/07/79      3:00AM
COMMAND? S
   KEY PARAMETERS PLEASE    M;MCLOGS

MCLOGS.HARRIS.COMPUTER
RUN DATE= 16/07/79  RUN TIME=      0  HRS
   PRIME SHIFT EXECUTION NOT PERMITTED
   JOB SCHEDULED FOR 22 HOURS
   JOB MODIFIED
COMMAND? S
   KEY PARAMETERS PLEASE    D;MCLOGS

MCLOGS.HARRIS.COMPUTER      DELETED
COMMAND? E

END OF PROGRAM
```

```
@
:STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979, 12:03 PM
COMMAND? B

   INPUT DETAILS OF JOB TERMINATED BY !EOJ

>!JOB TEST1,KIM.COMPUTER;INPRI=1
>!COMMENT SETUP TESTTAPE,WRITE/EXTERNAL/1600BPI
>!COMMENT THIS IS ONLY A TEST JOB
>!EOJ

   LIST,DELETE,REPLACE,KEEP,IMMEDIATE RUN,OR SCHEDULE?I

Q1971204.HARRIS.COMPUTER
RUN DATE= 16/07/79  RUN TIME=       0  HRS
Q1971204.HARRIS.COMPUTER     SUCCESSFULLY SCHEDULED
COMMAND? L

                      LBS SCHEDULER                     PAGE     1
  JOB          GROUP         ACCOUNT    RUN DATE       TIME     METRONET
 Q1971204     HARRIS       COMPUTER    16/07/79     SETUP=1
COMMAND? MODIFY

NAME OF FILE? Q1971204

GROUP?

ACCOUNT?

Q1971204.HARRIS.COMPUTER
IS THIS THE CORRECT JOB NAME (Y OR N) ? Y

DATE OF RUN?(DD/MM/YY)
ENTER HOUR OF DAY JOB IS TO BE RUN
BETWEEN 1 TO 24  OR ENTER 00 IF JOB CAN BE RUN AT ANY      TIME

RUN DATE= 16/07/79  RUN TIME=       0  HRS
IS THIS CORRECT? Y

   SETUP JOB SCHEDULED
   JOB MODIFIED
COMMAND? L

                      LBS SCHEDULER                     PAGE     1
  JOB          GROUP         ACCOUNT    RUN DATE       TIME     METRONET
 Q1971204     HARRIS       COMPUTER    16/07/79     SETUP=1
COMMAND? E

END OF PROGRAM
:
```

③

```
:STREAM


  JOB SCHEDULER .Version 03.40 .(c)1979 L.B.S. MON, JUL 16, 1979, 12:07 PM
COMMAND? B

   INPUT DETAILS OF JOB TERMINATED BY !EOJ

>!JOB KIM.COMPUTER;INPRI=2
>!COMMENT E[illegible]
>E
>!EOJ

   LIST,DELETE,REPLACE,KEEP,IMMEDIATE RUN,OR SCHEDULE?I

Q1971207.HARRIS.COMPUTER
RUN DATE= 16/07/79  RUN TIME=      0  HRS
SETUP JOBS MUST HAVE COMMENT SETUP CARDS
COMMAND? E

END OF PROGRAM
:
```

ADD NEW CAPABILITIES TO IMAGE/3000
BY CAPTURING TRANSACTIONS

ROBERT M. GREEN

ROBELLE CONSULTING LTD.
#130-5421 10th Avenue
Delta, B.C. V4M 3T9
Canada (604) 943-8021

ABSTRACT: This paper is addressed to the manager or designer of IMAGE/3000 on-line applications. I raise questions regarding throughput, integrity, response time, design ease, maintainability and failure recovery.

I suggest the use of an interface between the application programs and the database system to answer some of these problems. One such interface that I have developed is called SNAPSHOT, and is currently in use at a test site.

QUESTIONS FOR THE DATABASE MANAGER:

Q: Why should my on-line users have to wait at the terminal for non-urgent DBPUTs and DBDELETEs to complete processing?

Why should I have to artificially simplify my database design (by deleting paths) to provide on-line users with fast data-entry response times?

A: Let SNAPSHOT capture and defer instead.

WHAT IS DIRECT TRANSACTION PROCESSING?

Perform transactions on the database as they are collected from the on-line users. Since the update is done by the collection program, the user waits.

Presumably, other on-line users must be able to check this data at any time.

Lock records during updates for integrity.

Immediate update yields the slowest response time, but is the "normal" method (used by QUERY/3000 and most applications).

You capture the data, IMAGE organizes it. Usually, the organizing (indexing, sorting, posting to the disc) uses more resources than the collection.

WHAT IS DEFERRED TRANSACTION PROCESSING?

Gather and check data from on-line users, but merely capture the database changes for later update by a background utility program. This means throughput is increased.

Gather all data for a "GROUP" of related transactions before committing to final database changes. This means database consistency.

"HOLD" existing records until their updates are completed (perhaps hours later), even if the system crashes.

Trade off timeliness against response time and control. Do I really need that data in the database right now?

SNAPSHOT: CAPTURE NOW, ORGANIZE AT LEISURE

SNAPSHOT/ROBELLE is a software interface to IMAGE/3000 that takes care of capturing IMAGE/3000 transactions for deferred update.

Private workfiles for each user hold transactions against multiple databases. Transactions can be direct or deferred.

Background "UPDATE" program finishes deferred updates, and cleans up (i.e., completes or backsout) transactions after a crash.

Since Puts/Deletes take much longer than Updates (about a factor of ten), why not do Updates directly (to modify on-line status), and let the new records (Puts) follow in a deferred manner?

QUESTIONS FOR THE DATABASE MANAGER:

Q: What happens to my application programs if I change the structure of my database?

What if I want to use "@" in my calls to IMAGE to improve performance, but I also want data-independence?

What if I expand the Customer-Number and forget to change the buffers in my application program?

A: Let SNAPSHOT bind your programs to your databases at run-time.

RUN-TIME BINDING

An initial call to SNAPSHOT binds your program and field-lists to the database. You specify field-name-lists and set-names in a subprogram (using recoverable memory space).

SNAPSHOT checks the field-list and passes it to IMAGE. This allows you to use "*"-field-list and set-numbers in the rest of your program.

SNAPSHOT checks that the buffer size required for each dataset has not changed.

SNAPSHOT substitutes "@" whenever your field-list contains all fields of the set in the proper order.

Initial data binding provides full data-independence, in spite of database changes, and most efficient use of IMAGE by the program.

QUESTIONS FOR THE DATABASE MANAGER:

Q: How far from "standard" IMAGE will my programs wander if I use an interface package?

Who will write all of this initial run-time binding code (specifying field-lists and set-names)?

A: SNAPSHOT calls are so close to standard IMAGE calls that most code can be converted automatically with a few EDIT/3000 commands. Where special calls are needed (as in the initialization routine), SNAPSHOT provides a program to generate the source code automatically. It is no longer necessary to calculate the length of field-list buffers by hand; SNAPSHOT does this mechanical programming for you.

Q: Why should my system have to be down for hours in order to :RESTORE and roll-forward after every operating system failure?

Why should I have to worry nights about an application aborting and leaving the database contaminated?

A: Let SNAPSHOT capture and backout instead.

TERMINATIONS: FROM AMICABLE TO PSYCHOTIC

Normal program end - (sigh of relief)

Program bug - (when and how will I detect it?)

Program abort - (what was it doing?!)

System problem - (i.e., disc failure, but no crash)

System crash - warmstart (same as multiple aborts)

System crash - reload (transactions lost)

WHAT IS TRANSACTION LOGGING?

Uses "after" pictures of entries to roll-forward from a good backup copy after a crash.

Time-consuming; usually several hours.

Assumes a good backup copy of the database(s) and all transactions until the present. May fail if a transaction is lost.

Should leave the databases clean (i.e., no partially completed "groups" of transactions).

Most common reasons for rolling-forward are catastrophic system failure and operator (or program) error (database damaged).

WHAT IS TRANSACTION BACKOUT?

Need "before" pictures of entries to undo the transactions.

Used after a soft crash or program abort, instead of a full roll-forward; gets the system on-line faster.

Backout can be done by an application program itself (if it finds an irrecoverable error part way through a set), or by a stand-alone utility.

Records changed by the set must be "held" away from other users until the backout is done and the records are once again consistent.

WHAT IF I HAVE MULTIPLE DATABASES?

During a roll-forward (or backout) recovery, do I want to update the customer's balance if his sales order is not updated (in another database)?

Do my logging/backout methods recognize that transactions on different databases may be critically related?

If the answer is no, can I afford to merge all of my data into one IMAGE/3000 database (255 items!)?

THE PROBLEM OF INTERVENING UPDATES

What has happened to this customer in the database since I retrieved him and displayed him on the screen for my operator to change?

By keeping the record (or database) locked, I can be sure that nothing happened; but it may stay locked for a long time.

Why couldn't the system tell me if anyone had changed my records since I retrieved them? If I can detect intervening changes to the entries, then I can cancel and redo my own.

FAIL-SOFT RECOVERY DESIGN

Eliminate heavy dependence on record numbers, both in masters (migrating secondaries) and details (lost transactions).

Eliminate heavy dependence on the time-sequence of updates. Within a transaction workfile, each "group" of transactions is unrelated to the others and can be completed even if the others fail.

"Hold" records to prevent updates by other users, even if the holding user aborts or the system crashes. Stronger than a lock.

After a crash, the databases and transaction files must be cleaned up before full processing can be allowed to continue.

SNAPSHOT SURVIVAL PACK

Backout of "direct" transactions via the workfile.

Archiving and auditing transactions.

You can lock a set of records even across system crashes.

Handles transactions for multiple databases as a single "group", not as separate "groups".

Intelligent crash recovery.

QEDIT: A SYSTEM LOAD ANALYSIS

ROBERT M. GREEN

ROBELLE CONSULTING LTD.
#130-5421 10th Avenue
Delta, B.C. V4M 3T9
Canada (604) 943-8021

ABSTRACT: The system load of any program on the HP 3000 is composed of cpu time consumed, disc accesses demanded, data stack size and code segment number, size and frequency of transfer. QEDIT is an attempt to make on-line programming more practical by reducing the cost and customizing editing features for programmers.

Many HP 3000 users underestimate the resources consumed by on-line programming. What does a programmer go through to accomplish a simple, one-line change to a program on the HP 3000?

> Using EDIT/3000, the programmer "texts" a copy of the program into the editor workspace, then modifies a line and "keeps" another copy so that it can be compiled.
>
> Assuming a 3200-line source file, this innocent-sounding cycle will take from 3 to 10 minutes (depending upon the system load) and will consume these resources:
>
> 76 c.p.u. seconds
>
> 1236 d-i-s-c transfers

What is the effect on the system response time while this program change (TEXT/KEEP) is occurring in the computer? Take a look at an actual HP 3000 in commercial usage:

> Before the program change begins, the response time on this 512K-byte system is immediate (less than a second) for inquiries into a database.
>
> During the time of the change, the response time on all terminals degrades to 2 seconds.
>
> Now put three programmers on the system and let them do their changes concurrently: response time stretches out to 6 seconds.

What options does the system manager have to restore response times to their former level?

> Force programmers to work nights and weekends?
>
> Upgrade to a larger computer?

Or, the system manager can try another method of on-line programming that eliminates the TEXT/KEEP cycle. With QEDIT, instead of using TEXT/KEEP, the programmer OPENs his "source" file (maintained in a special format) and modifies it directly. The same one-line program change (referred to on the previous page) can be done in seconds, consuming only 2 c.p.u. seconds and less than 10 disc transfers.

User response times are not degraded by on-line programming. Switching a typical installation to QEDIT releases the computer power to add more user terminals, run an extra batch job, or give every programmer a personal terminal.

In addition to QEDIT's economical use of c.p.u. time and disc transfers, here are examples of other measures of system load:

Using "OVERLORD", from the Contributed Library, you can compare the data stack sizes of QEDIT and EDIT/3000:

QEDIT: 3600 words

EDIT/3000: 6300 words

Using "PROGSTAT", from the Contributed Library, you can compare the code segment sizes of QEDIT and EDIT/3000:

QEDIT: 7 segments, 15600 words

EDIT/3000: 12 segments, 33400 words

QEDIT is about 1/2 the size.

An actual comparison of disc space using a 3200-line source file:

EDIT/3000: 1025 sectors for a "KEEP" file.

PLUS 1800 sectors for a working copy.

TOTAL 2825 sectors for one program.

QEDIT: 390 sectors for a source file.

PLUS 0 sectors for a working copy (not needed).

TOTAL 390 sectors of disc consumed.

SAVINGS: 2435 disc sectors for ONE program.

A Distributed Structural Design Graphics System

Lawrence E. Nolan

Introduction

The FASTDRAW/3 distributed graphics system includes commands to generate points and elements in local coordinate systems, perform simple construction of arcs, circles and create arbitrary space curves and surfaces. FASTDRAW/3 can automatically create a mesh of elements on a defined surface after control points along the boundaries have been selected. Also, elements can be duplicated by any combination of translation, rotation, and scaling in two or three dimensions. FASTDRAW/3 supports user defined elements that allow a design to be accomplished in parts and the parts then combined to build the final geometric model.

The FASTDRAW/3 distributed processing system is compared with alternative systems on large mainframe computers, standalone mini-computers, and desktop computers. The distributed processing system is part of the MCAUTO network of remote job entry stations. It is a mini-computer implementation of FASTDRAW/3 with enhancements to provide better service. Local plotting from a high speed pen and ink plotter that is directly attached to the mini-computer gives the engineer fast turnaround on plots of his design at intermediate steps. Error free communications between the user's graphic CRT and the FASTDRAW/3 system occurs at speeds higher than available from conventional remote timesharing services.

The structural design engineer today has a choice of computer programs and systems which are supposed to reduce the work required to create the input data used by finite element analysis programs such as STRUDL, EASE, ANSYS, etc.

The manner in which the engineer uses these preprocessing systems varies from one program to another. Some are simple prompting programs that create card files to be scanned and subsequently plotted on a graphics CRT.

---

*SUPERVISOR, Graphic Products - McDonnell Douglas Automation Company

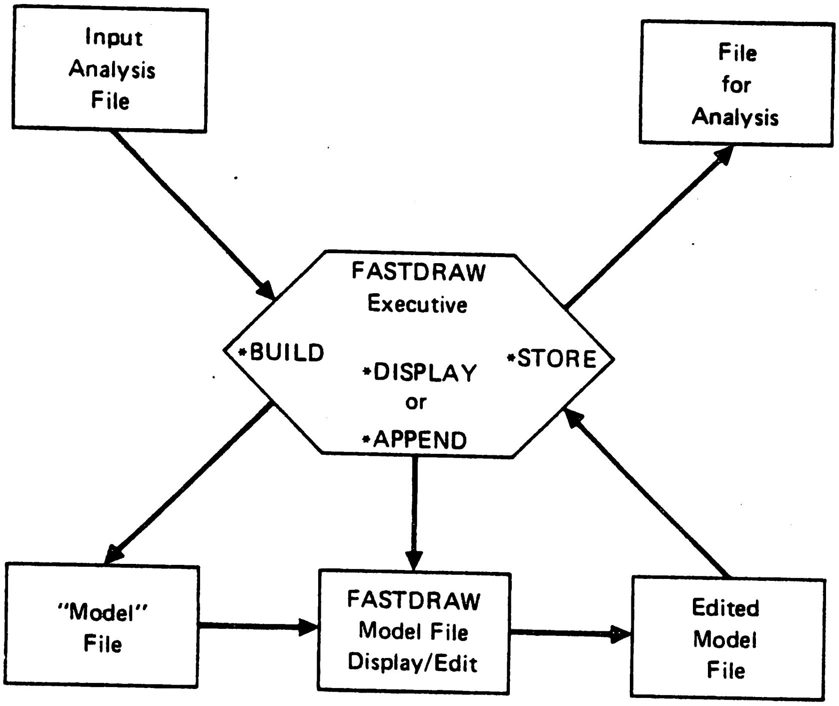
These prompters are usually specific for a single analysis input format and are cumbersum if used in an interactive design process. The programs that allow interactive design and editing do not all satisfy reasonable expectations for ease of use, understandable conventions, and good response.

This paper describes the FASTDRAW/3 interactive graphics system as it is implemented in the McDonnell Douglas Automation Company distributed processing network.

The system provides interactive finite element design pre and postprocessing as well as access to analysis programs such as STRUDL, NASTRAN, ANSYS, EASE, SAP which execute on MCAUTO's large IBM and CDC remote job entry computers.

FASTDRAW/3 Features

FASTDRAW/3 allows the finite element designer to create, display, and modify a special data file known as the model file. The interactions with the program are performed from an interactive graphics CRT. The FASTDRAW/3 Executive (see Fig. 1) interfaces the engineer with the model manipulation command subsystem and with the conversion interfaces for STRUDL, NASTRAN, ANSYS, EASE2, and SAP5.


Input Analysis File
File for Analysis
FASTDRAW Executive
•BUILD
•DISPLAY
or
•APPEND
•STORE
"Model" File
FASTDRAW Model File Display/Edit
Edited Model File

FASTDRAW/3 ORGANIZATION
Figure 1

1.1 The FASTDRAW/3 Model File

Using FASTDRAW/3 in finite element design consists of creating and modifying data kept in the model file. A model file contains points, local coordinate system definitions, elements, boundaries, and regions. Let's briefly descirbe each of these.

Point data consists of the x,y,z coordinates, a reference to a local coordinate system, an identifying label, force constraints, a count of the number of elements that use the point and the number of times the point is used by boundaries. Points attached to a boundary are called control points because they control the mesh that is generated on a region formed with that boundary.

A local coordinate system has a label, the coordinates of its origin, and a type(rectangular, cylindrical, or spherical).

FASTDRAW/3 elements are either finite elements specific for an analysis program or are user-defined elements. A user-defined element is constructed in its own model file, and consists of application finite elements. In another model file, instances (incidences) of the entire user-defined element can be created by locating the definition points to be used in the new model, and by also indicating which user-defined element is required.

Boundaries are used to create a mesh of points and elements on a region. A boundary can be a straight line, a circular arc, or a space curve. Points are created along a boundary to influence the generation of a mesh and in some cases to aid in the creation of additional boundaries. A temporary boundary consisting of several other connected bounaries can be specified with the PATH command.

The last FASTDRAW/3 data type is the region. A region is a surface or part of a surface. The region is speified by three or four sides which are one or several boundraies. The region is used to generate a mesh of finitie elements such that the points in the mesh are on the region.

The values of these data types is easily printed for inspection on the CRT by using the PRINT command.

1.2 Command Usage.

Commands will usually be shown with their optional parameters. When a required parameter is left off of the command line, FASTDRAW/3 will request the user to enter the missing parameter values. This is an important feature for someone unfamiliar with a specific command.

When using the FASTDRAW/3 model file editor all commands are always available.

What this means is that the user is not required to make command selections that limit what operations can be done at a particular stage of the design.

1.3 Label, PRINT Command

The first example is the LABEL command. It is used to print the labels of model file data entities which are currently displayed. Its full syntax is:

```
LABEL type[(list)],[type...] [REACTIONS][MESH]
```

LABEL will label Points, elements, regions, systems, and boundaries, or all types of data as well as reactions(i.e., constraints) and meshs. If no parameters are specified, then the data type choices are printed on the CRT and the user selects one of the types. FASTDRAW/3 will then label all of the occurences of that type in the picture. An example of LABEL which will label all the regions, and the points numbered between 10 and 50:

```
LABEL R,P(10-50)
```

The values of the FASTDRAW/3 data types that were described in the preceeding section can be displayed very simply by using the PRINT command:

```
PRINT [type[(list)]]... [PATH] [SWITCH]
```

So, to print the model file's point data, this command would be used:

```
PRINT P
```

1.4 Display Control

FASTDRAW/3 has several commands which control the display of the model file. They do not change values of the model data, only change the picture that is visible on the CRT. Two of these commands, LABEL and PRINT, have already been described.

The REORIENT command lets the user select three viewing angles which causes the picture to be redrawn in the desired orientation:

```
REO ANG(thetaz,thetay,thetax)
```

If no angles are specified, the command prompts the user to enter the angles.

Portions of the picture can be magnified for closer inspection(zoom) by using the WINDOW command. WINDOW lets the user pick two points from the model or two points on the CRT screen(using a cross-hair cursor) which define opposite corners of a rectangle that is to be enlarged to fill the CRT screen.

2.0 Model file Data Creation

The different model file data types (points, elements, systems, regions, boundaries) each have their own set of commands that are used to construct new occurrences of these data types. In addition, points and elements can be duplicated using four special duplication commands. There is also a command to generate a mesh of points and elements on a region.

2.1 Creating Points

There are nine commands that will create points in the model file. These are:

POINT DIGITIZE
POINT at the intersection of 2 boundaries
POINT KEYIN
POINT HORIZONTAL
POINT VERTICAL
POINT on a BOUNDARY
POINT on a PATH
POINT on a REGION

Keyed-in points may be specified in a user-defined local coordinate system (rectngular, cylindrical, or spherical). As an example, if local system number 1 is a spherical system, the FASTDRAW/3 command

```
POI KEY SYS(1)
```

would prompt by printing

```
ENTER   R,THETA,PHI
```

Then coordinate triples could be entered, with an empty line terminating the input.

The POINT ON A BOUNDARY command has five variations which offer different methods for automatically calculating the distances between the new points that are placed on the selected boundary.

2.2 Creating Elements

FASTDRAW/3 uses the same commands to create both user-defined elements and application finite elements. Elements can be constructed one at a time or by using the duplication commands or the MESH command.

A FASTDRAW/3 user selects which application or user-defined elements are going to be used and assigns the element a number. To add a single element instance to the model file the ELEMENT command is used. The required definition points are specified by their label or by digitizing each point on the CRT screen

```
ELEMENT BOX(n) P(point list)
   or ELn P(point list)
```

(where "n" is the number that was assigned to the element by the user).

For example, EL2 P(1,7,10,4) could be used to add a new occurrence of the element assigned to box 2 (it has four definition points).

This is the manual method of creating elements.

There are also four element and point generation commands that can create new elements and points from existing points and elements:

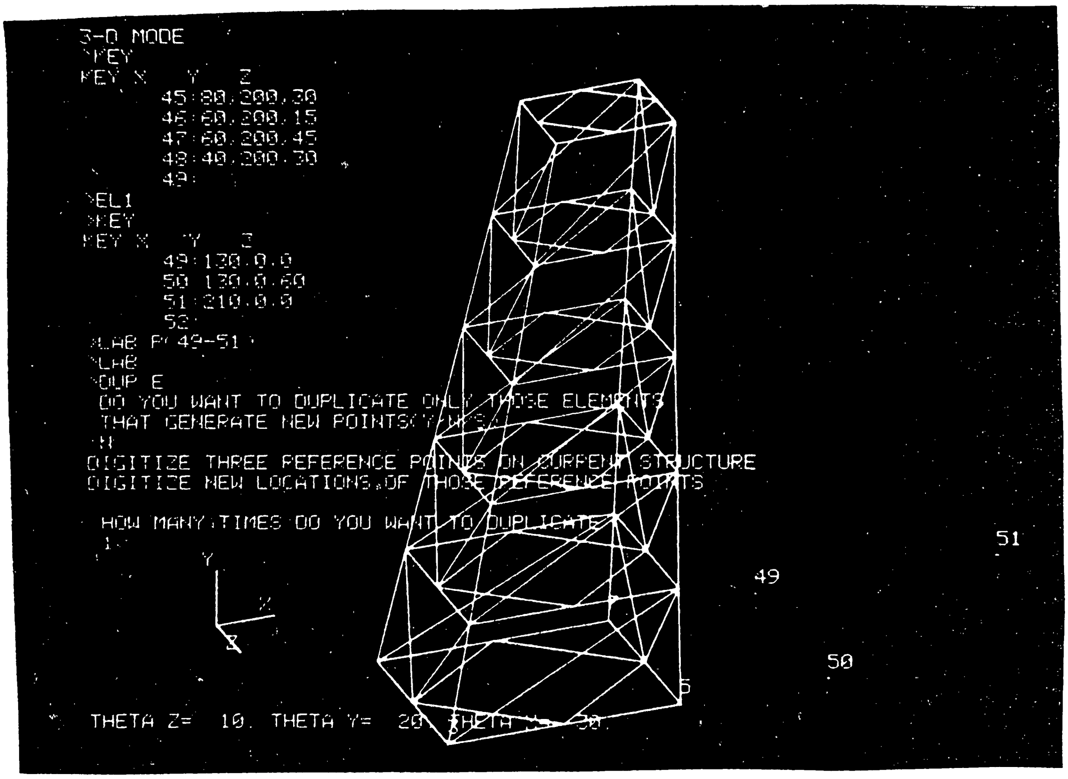
TRANSLATE
ROTATE
TRAVERSE
DUPLICATE

The TRAVERSE command generates new elements by translating points in the same plane or in a plane parallel to the current plane.

The ROTATE command duplicates the desired elements by rotating the new elements a specified number of degrees about an axis lying in the current plane, or one perpendicular to the current plane.

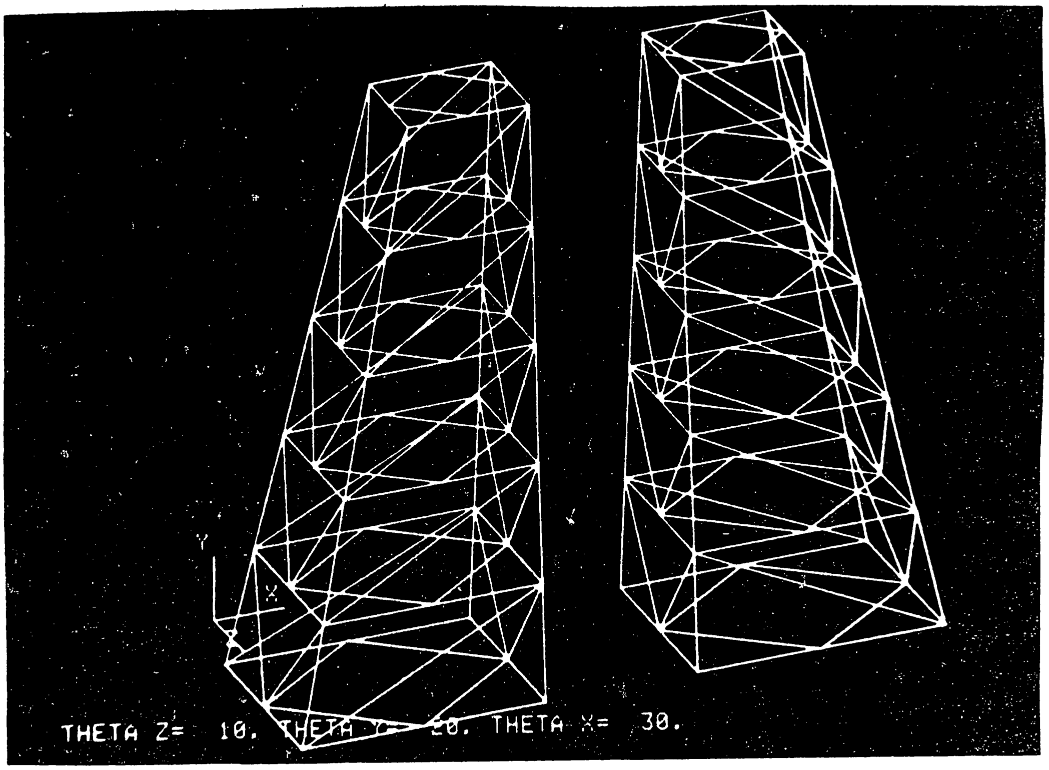
The TRAVERSE command is specifically designed for truss generation because it generates the duplicates such that the reference lines in the new elements have the same slope as in the original.

The DUPLICATE command works in three dimensions and combines translation, rotation and scaling for creating duplicates. (see Figures 2, 3)



```
3-D MODE
>KEY
KEY X    Y    Z
        45:80,200,30
        46:60,200,15
        47:60,200,45
        48:40,200,30
        49:
>EL1
>KEY
KEY X    Y    Z
        49:130,0,0
        50 130,0,60
        51:210,0,0
        52:
>LAB P 49-51
>LAB
>DUP E
 DO YOU WANT TO DUPLICATE ONLY THOSE ELEMENTS
 THAT GENERATE NEW POINTS(Y/N)?
>N
DIGITIZE THREE REFERENCE POINTS ON CURRENT STRUCTURE
DIGITIZE NEW LOCATIONS OF THOSE REFERENCE POINTS

 HOW MANY TIMES DO YOU WANT TO DUPLICATE
 1

 THETA Z=  10. THETA Y=  20. THETA X=  30.
```

MODEL BEFORE ELEMENT DUPLICATION
Figure 2



```
THETA Z=  10. THETA Y=  20. THETA X=  30.
```

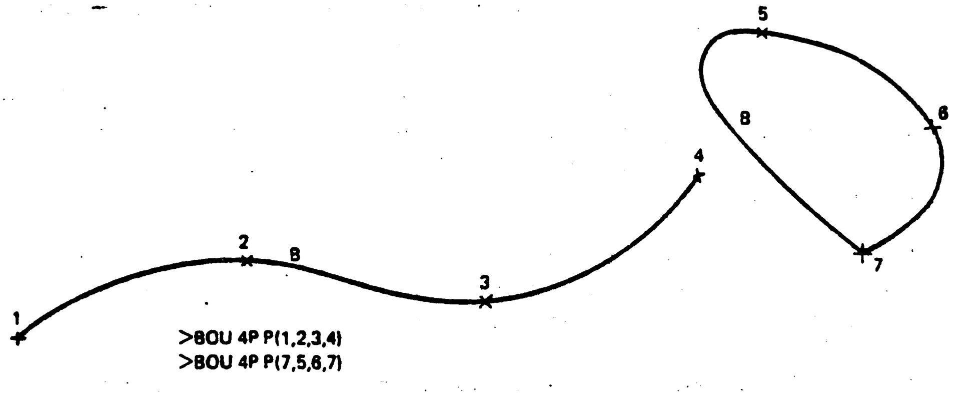
MODEL AFTER ELEMENT DUPLICATION
Figure 3

2.3 Creating Boundaries

There are fifteen boundary creation commands. These commands create standard geometric boundaries:

LINE P	line, given a point and an angle
LINE NP	line through a series of points
LINE LINE	line parallel to another line
ARC 3P	circular arc through 3 points
ARC 2L	arc tangent to 2 lines
ARC 3L	arc tangent to 3 lines
CIRCLE P	circle, center point and radius
CIRCLE 2P	circle, center, circumference pt.
CIRCLE 3P	circle, given 3 points
BOUNDARY 4P	space curve through 4 pts.
BOUNDARY SPLIT	two boundaries by cutting the existing one
BOUNDARY REGION	boundary string through points on a region
BOUNDARY NP	smooth boundary through points

The BOUNDARY 4P command will create a cubic space curve through four model file points (See Figure 4).



```
>BOU 4P P(1,2,3,4)
>BOU 4P P(7,5,6,7)
```

BOUNDARY FROM 4 POINTS
Figure 4

The BOUNDARY NP command uses a series of model file points to create a continuous spline curve. The mathematical technique used by FASTDRAW/3 to represent boundaries is called parametric cubic geometry. (See Ref. 2)

When describing a complex shape with FASTDRAW/3 several different boundaries may be needed to model the geometry of the object.

The PATH command: PATH B(boundary list) lists a set of boundaries that together define a continuous boundary chain. The path is used in creating points along the path, or in creating a region (surface) that follows the path (see REGION PATH).

2.4 Creating Regions(Surfaces)

There are five commands used to create regions with FASTDRAW/3:

```
REGION PATH B(boundary list)
REGION SPHERE RADIUS(r) THETA(tba,tea)
              PHI(pba,pea) SYSTEM(s)
REGION SUBDIVIDE R(r1) B(b1) B(b2) B(b3) B(b4)
REGION 4B B(b11) B(b2) B(b13) B(b4)
REGION RULED B(b11) B(b12)
```

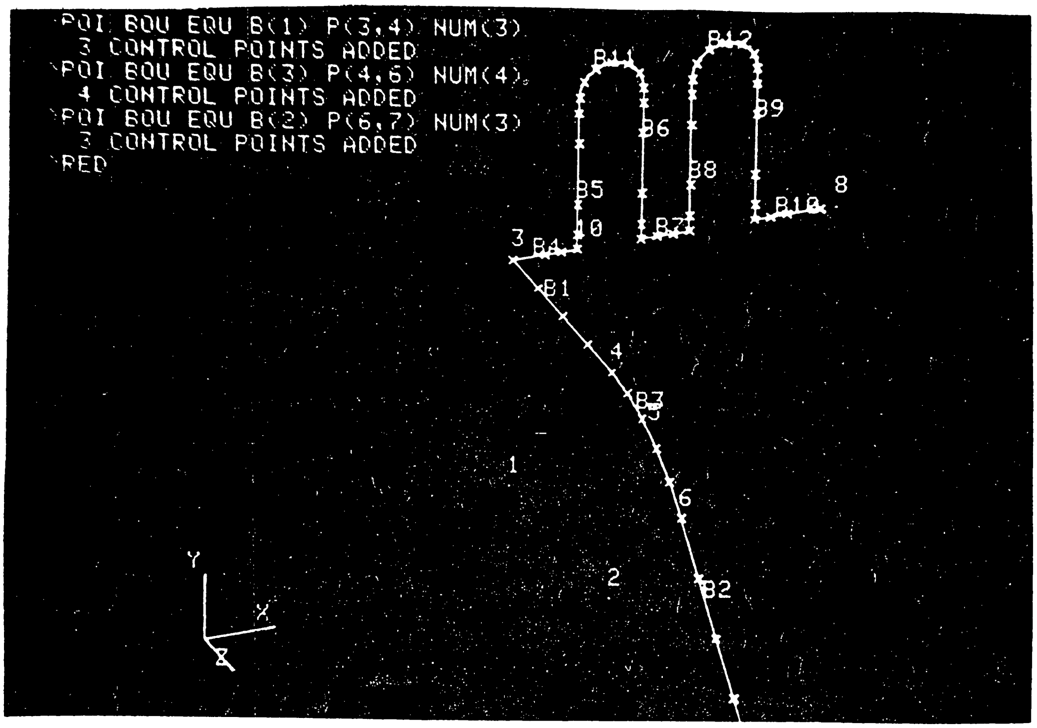
REGION SPHERE is a command to create a spherical surface given the radius and the starting and ending theta and phi angles.

REGION RULED creates a ruled surface by using straight lines to connect two boundary lists that form the opposite edges of the surface.

REGION SUBDIVIDE creates a smaller surface from an existing one. Only the limits defining the region change. Typically, REGION SUB is used to form a smaller region for mesh generation.

REGION PATH is used when a surface is desired that can be created by a curve of constant shape which is moved along the specified path.(Figure 5, 6)

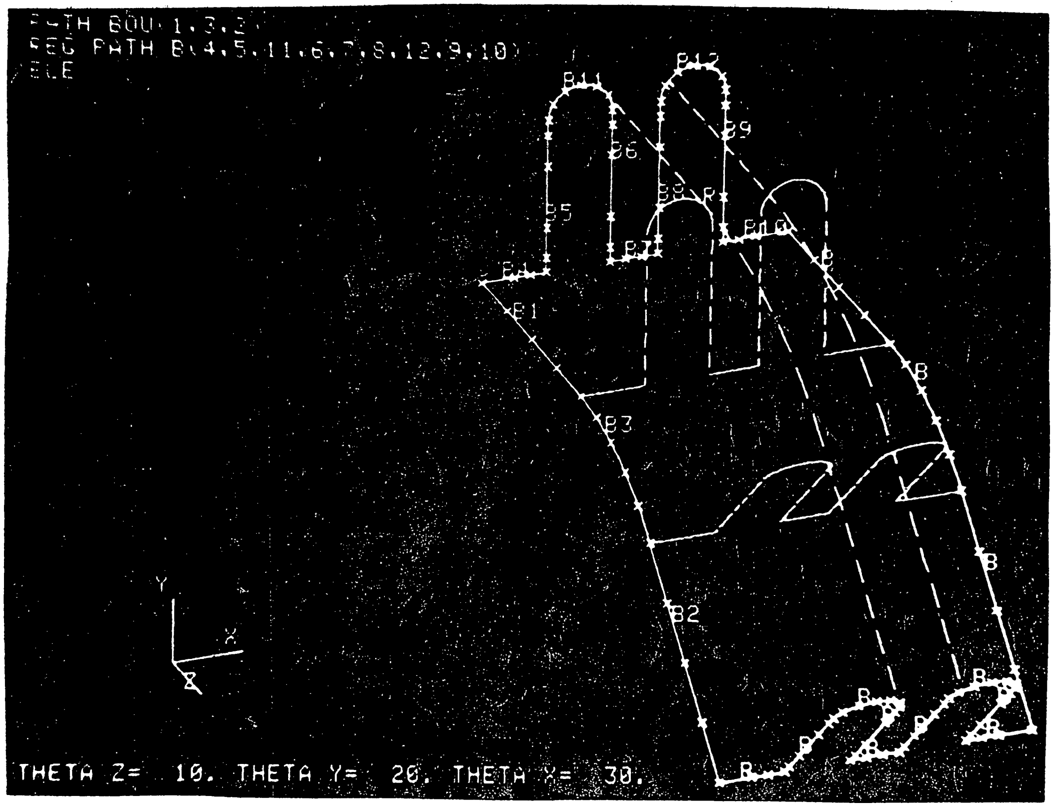
REGION 4B is used in creating a region from four boundary lists.



```
POI BOU EQU B(1) P(3,4) NUM(3)
 3 CONTROL POINTS ADDED
POI BOU EQU B(3) P(4,6) NUM(4)
 4 CONTROL POINTS ADDED
POI BOU EQU B(2) P(6,7) NUM(3)
 3 CONTROL POINTS ADDED
RED
```

B11
B12
B9
B6
B8
8
B5
B10
10
B7
3
B4
B1
4
B3
5
1
6
2
B2
Y
X
Z

CREATION OF CONTROL POINTS
Figure 5



```
PATH BOU(1,3,2)
REG PATH B(4,5,11,6,7,8,12,9,10)
ELE
```

```
THETA Z=  10. THETA Y=  20. THETA X=  30.
```

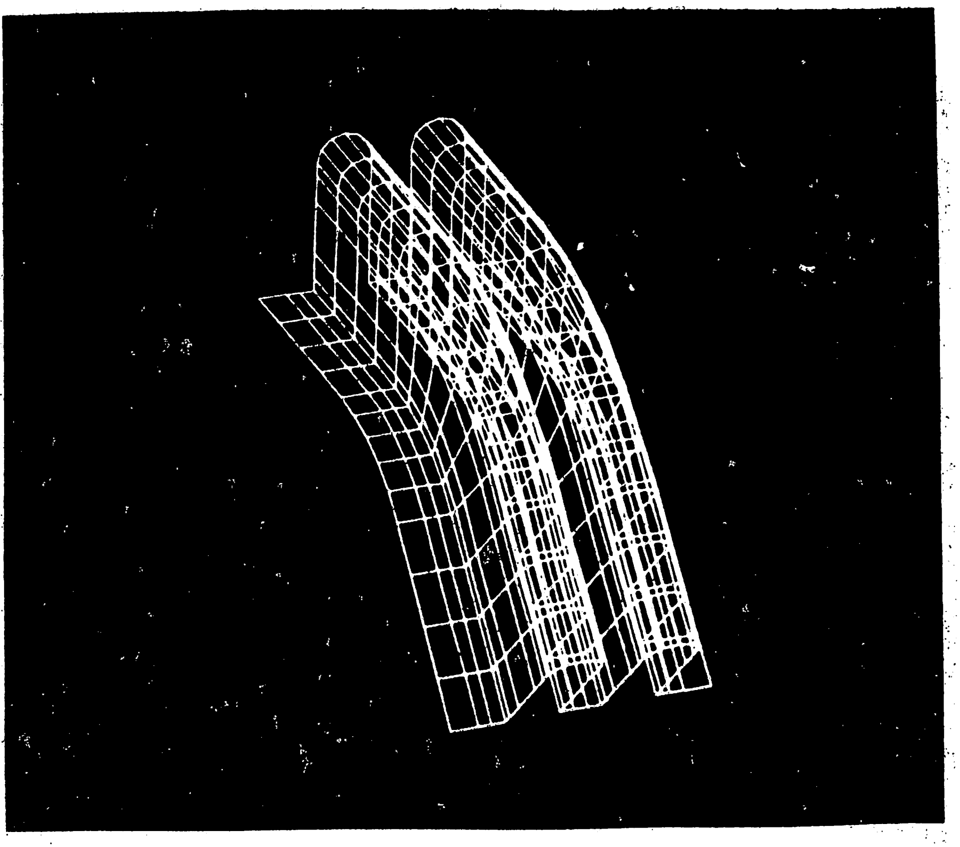
B11
B12
B9
B6
B8
B10
B5
B7
B4
B1
B
B3
B
B2
B
B
B
Y
X
Z

REGION PATH SURFACE CREATION
Figure 6

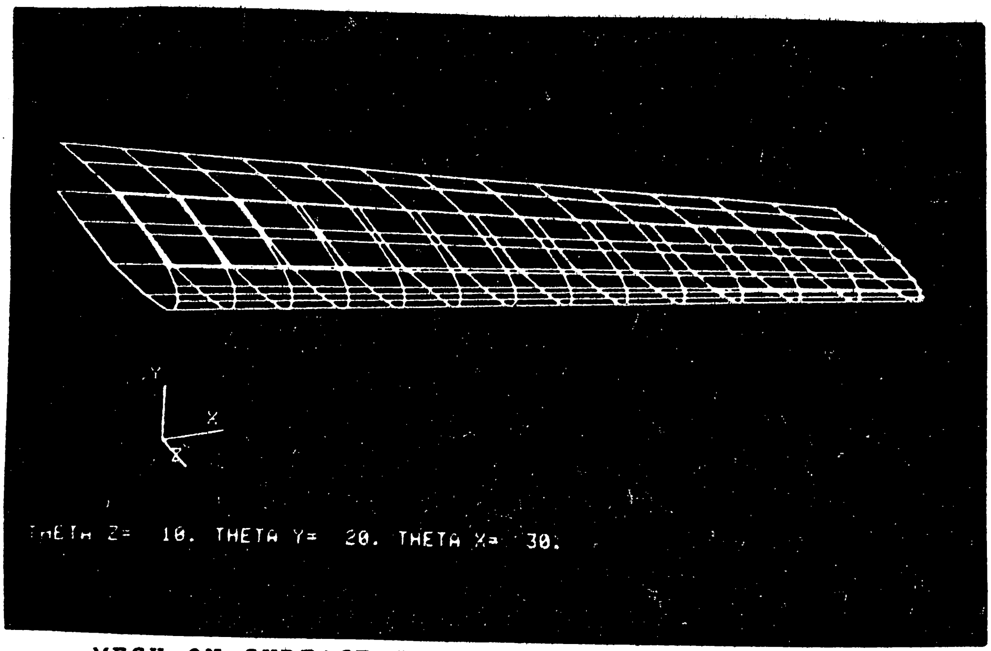
2.5 Creating a Mesh

The MESH command is used to generate a finite element mesh on any FASTDRAW/3 region. Control points placed along the region's boundaries with the POINT creation commands are the way that the user has to adjust the mesh density. A separate command, MSET, is used to control the label generation sequencing and the type of mesh elements to be used. FASTDRAW/3 will automatically create transition row elements if the proper number of control points are present on opposite sides of the region.

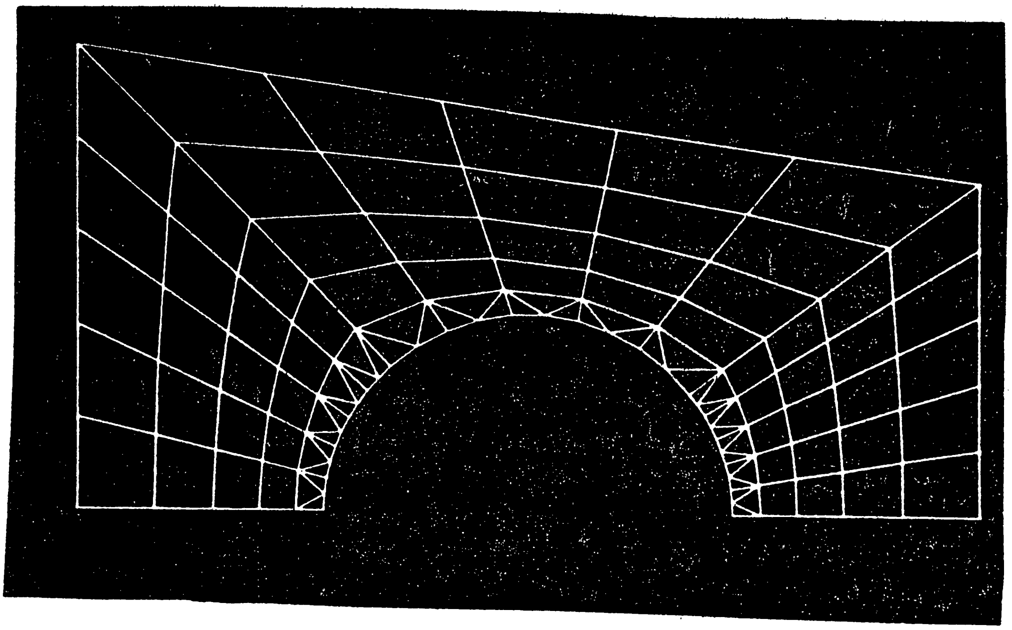
The optimum use and selection of boundaries and regions for mesh generation is extensively covered in "FASTDRAW/3: Interactive Graphic Mesh Generation" (Reference 3). (Examples of meshs produced by FASTDRAW/3 are shown in Figures 7, 8, 9.)



MESH ON SURFACE AFTER REGION PATH
Figure 7


Y
X
Z
THETA Z= 10. THETA Y= 20. THETA X= 30.

MESH ON SURFACE FORMED BY REGION RULED
Figure 8



MESH WITH TRANSITION ROWS
Figure 9

3.0 Distributed System Design

All the display and model editing features of FASTDRAW/3 have been incorporated in a multiple user mini-computer system. This implementation strategy was chosen over several other possibilities:
- timeshared large mainframe
- single user mimi-computer
- desktop computer

FASTDRAW/3 was already available on MCAUTO's timeshared system and it was desired to reduce the cost of the computer system that hosted FASTDRAW/3, and to also provide service enhancements that were difficult to do on the CDC equipment being used.

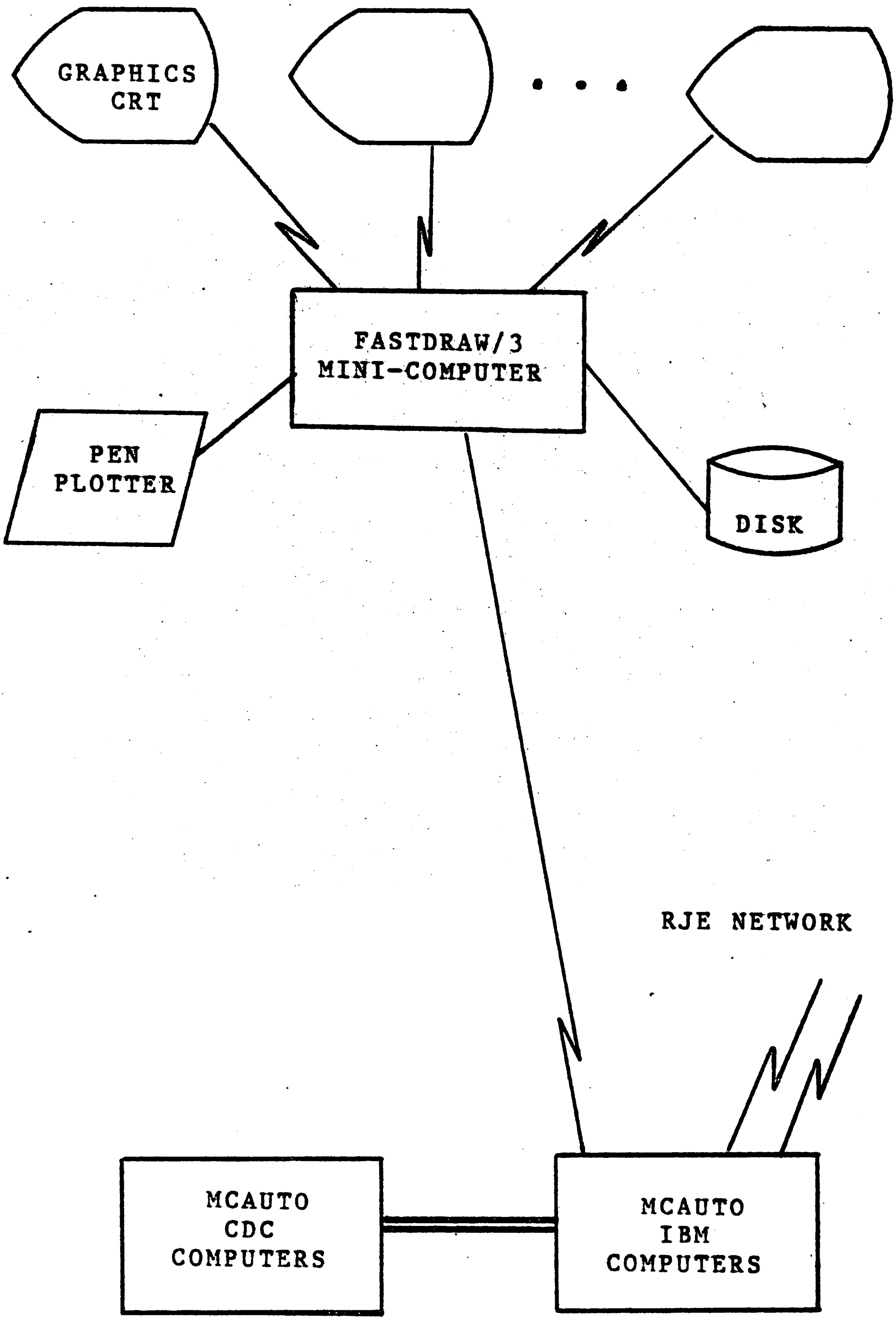
Small, single user mini-computers had several strikes against them in spite of their low cost. The small memory space on these mini's would have made it difficult to implement FASTDRAW/3. Another important consideration was that important peripherals (disks, plotters) are a significant fraction of the cost of the system. To keep the cost low for a user's workstation, these devices should be shared and easily accessable. A final point to be mentioned is that to ask a customer to purchase their own mini-computer imposses a large capital expenditure and one which is quickly obsoleted or at least soon surpassed in value by new equipment.

The last strategy considered was to use a desktop computer. All of the implementation problems found with small mimi-computers existed with desktop systems as well. The expense of non-shared peripherals was also still a problem.

What finally evolved was a multi-user timeshared mini-computer. The restriction on memory space was overcome by using a mini-computer with virtual memory management hardware and software. This virtual memory allows for large programs and sharing of a single copy of a program by all users.

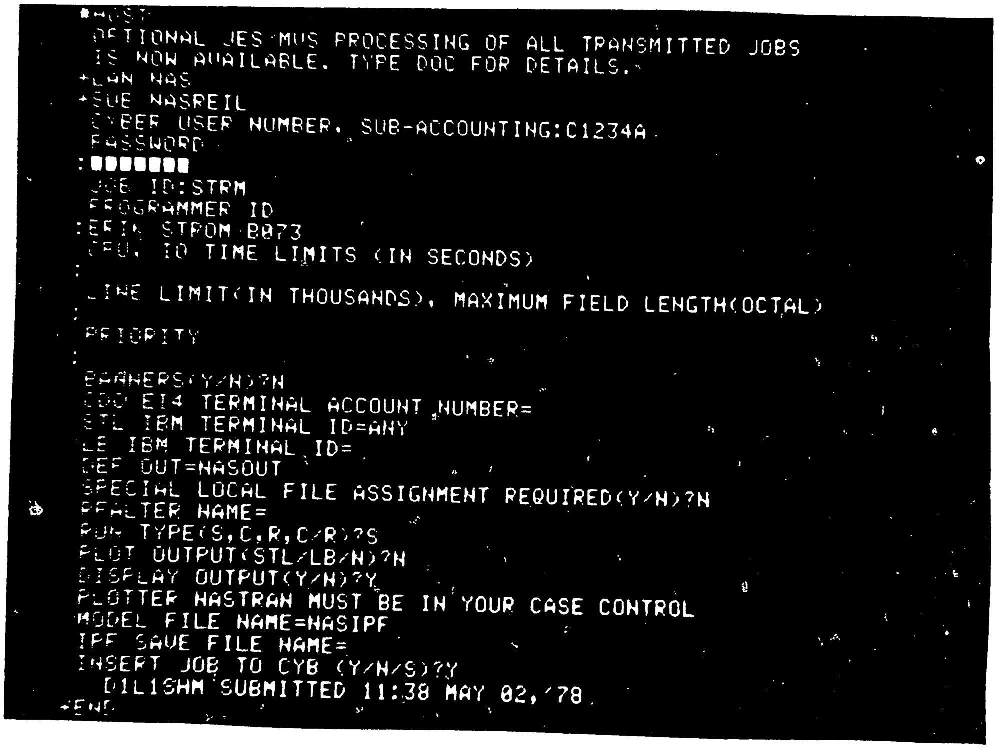
The distributed FASTDRAW/3 system does not rely on any special operating system modifications. This keeps the software portable and makes it possible to respond to improvements in hardware cost or performance as they become available.

The FASTDRAW/3 distributed processing system is a node in the MCAUTO network of remote job entry terminals and computers. (see Figure 10) The multi-user mini-computer allows several FASTDRAW/3 users to access the FASTDRAW/3 model editing features while the computer system is concurrently processing plots spooled to an on-line pen and ink plotter, and also communicating to the MCAUTO RJE complex of IBM and CDC computers.


GRAPHICS CRT
FASTDRAW/3 MINI-COMPUTER
PEN PLOTTER
DISK
RJE NETWORK
MCAUTO CDC COMPUTERS
MCAUTO IBM COMPUTERS

DISTRIBUTED PROCESSING SYSTEM NETWORK
Figure 12

FASTDRAW/3 has a subsystem called HOST that makes the communication between the mini-computer and the MCAUTO IBM and CDC systems a simple matter of answering questions. (Figure 11) HOST then creates the correct batch job control card images and submits them to the proper host computer in the MCAUTO network.



```
*HOST
 OPTIONAL JES/MVS PROCESSING OF ALL TRANSMITTED JOBS
 IS NOW AVAILABLE. TYPE DOC FOR DETAILS.
*LAN NAS
*SUE NASREIL
 CYBER USER NUMBER, SUB-ACCOUNTING:C1234A
 PASSWORD
:■■■■■■■■
 JOB ID:STRM
 PROGRAMMER ID
:ERIK STROM B073
 CPU, IO TIME LIMITS (IN SECONDS)
:
 LINE LIMIT(IN THOUSANDS), MAXIMUM FIELD LENGTH(OCTAL)
:
 PRIORITY
:
 BANNERS(Y/N)?N
 CDC EI4 TERMINAL ACCOUNT NUMBER=
 STL IBM TERMINAL ID=ANY
 LB IBM TERMINAL ID=
 DEF OUT=NASOUT
 SPECIAL LOCAL FILE ASSIGNMENT REQUIRED(Y/N)?N
 PRALTER NAME=
 RUN TYPE(S,C,R,C/R)?S
 PLOT OUTPUT(STL/LB/N)?N
 DISPLAY OUTPUT(Y/N)?Y
 PLOTTER NASTRAN MUST BE IN YOUR CASE CONTROL
 MODEL FILE NAME=NASIPF
 IPF SAVE FILE NAME=
 INSERT JOB TO CYB (Y/N/S)?Y
   D1L1SHM SUBMITTED 11:38 MAY 02,'78
*END
```

EXAMPLE OF HOST OPERATION
Figure 11

4.0 Summary

FASTDRAW/3's superior features such as local coordinate systems. flexible point creation, element duplication, and boundary, region and mesh generation have been enhanced by implementing FASTDRAW/3 on a multi-user mini-computer system. Now on-line pen and ink plotting of intermediate and final design steps can be performed with the plotted output quickly available. Terminal drawing speeds 2-8 times faster than what is available from remote timesharing mainframes is a standard feature. A successful distribution of tasks has been accomplished by placing the interactive problem modeling software on a shared mini-computer that has a link to the MCAUTO large scale batch processing system to execute the model analysis.

References

1. FASTDRAW/3 Reference, McDonnell Douglas Automation Company, 1978.

2. Fousek, Daniel P., "An Interactive Mesh Generation System", presented at the ASCE Annual Convention, Chicago, Illinois; October 1978.

3. Fousek, Daniel P., "FASTDRAW/3: Interactive Graphic Mesh Generation", to be presented at the Seventh Conference on Electronic Calculation, ASCE, St. Louis, Missouri; August 1979.

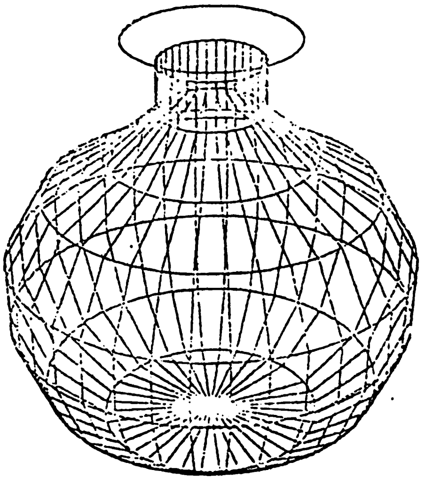
Acknowledgements

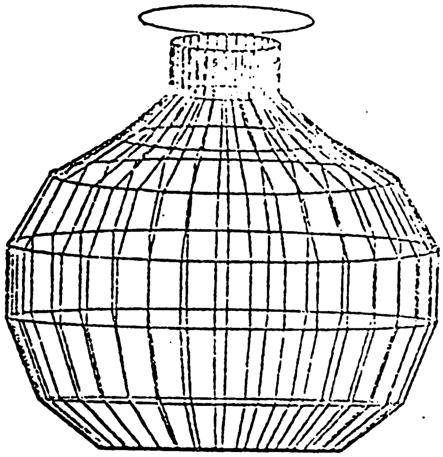
This section lets the author recognize the fine work done by those involved in the design, implementation, and support of FASTDRAW/3 and the distributed system:

Larry Barnes
Pat Giarritano
Jill Holtz
Ned Hurst
Kathy Kumpis
Ted McFadden
Debbie Scovill
Douglass Bates
Donn Gunter
Janina Kozacka
Robert Linback
James Mohan
Laila Robinson
Paul Sicking

GPGS-F, A PORTABLE DEVICE INDEPENTANT GRAPHIC SYSTEM.

Authors : Fredrik Major, Ditlef Martens
The Ship Research Institute of Norway
(NSFI)





The Ship Research Institute of Norway (NSFI) has implemented GPGS-F on the HP-3000 series computer and is responsible for HP-versions of the system.

GPGS-F is a system near to the standards for graphic systems proposed by standardisation organizations in several countries and is choosen as standard graphic system by the Norwegian Co-operation in Computer Graphics (NORSIGD). This organization is responsible for the basic development of GPGS-F which is carried out on a Univac 1100 series computer by the Computing Centre at the University of Trondheim (RUNIT).

The system is implemented on various diffirent computers and has device drivers for many of the common graphic devices. In this paper GPGS-F will be described from a functional point of view, aiming to give potential users of the system an overview of it's capabilities.

1. HISTORY

The GPGS system was originally designed by Rekencentrum, Delft University of Technology, The Netherlands and Science Faculty, Catholic University Nijmegen, The Netherlands in 1972. A version of the system written in standard Fortran has been developed by NORSIGD. The first Fortran based version was released in 1975 and named GPGS-F. GPGS-F has been under continous development since the first version was released. This work has been guided by annual user meetings. The result of these meetings has been new features and minor changes to the system.

2. DEVICE CONTROL

GPGS-F provides device independent programming with choice of graphic device at run-time.

Figure 1 shows how the device independancy is obtained. The device independant part produces the same picture code for all devices, and the device driver(s) translates this code to the bit pattern required for the actual device. The device independant code is put on such a level that advanced graphic devices may be used in an efficient way. Examples of such functions are character, circle and marker generation. GPGS-F will also be able to take advantage for the functions of more advanced refresh display such as hardware scaling, rotation and depth quing. The graphic devices currently supported in the HP-3000 version og GPGS-F are:

Tektronix 4010/4014
Tektronix 4662
Calcamp plotters
HP 7221 plotter
HP 2648 graphic screen

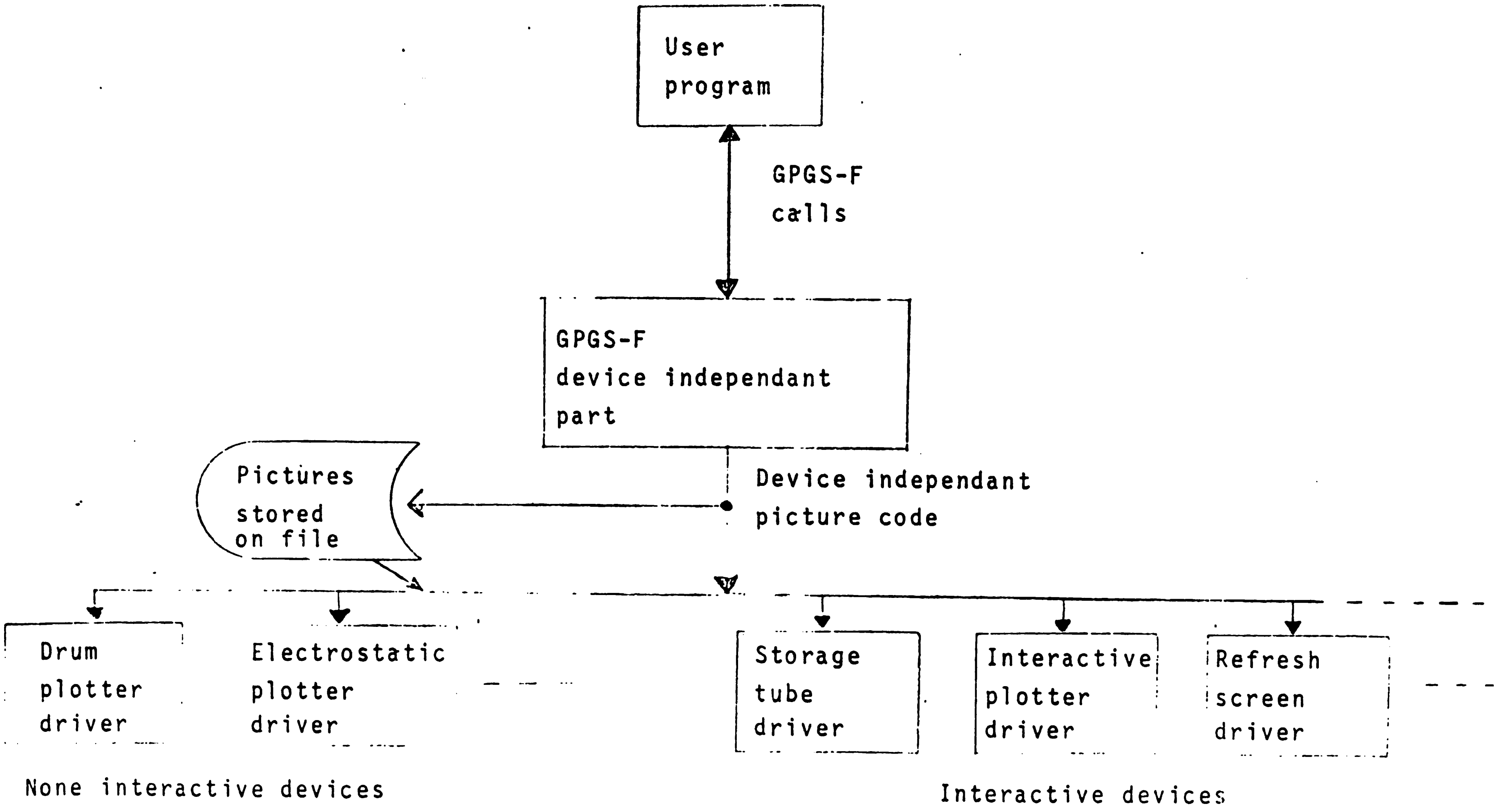

User program
GPGS-F calls
GPGS-F device independant part
Device independant picture code
Pictures stored on file
Drum plotter driver
Electrostatic plotter driver
Storage tube driver
Interactive plotter driver
Refresh screen driver
None interactive devices
Interactive devices

Fig. 1. MAIN STRUCTURE OF GPGS-F.

3. GRAPHIC ELEMENTS.

By single calls to the basic routines of GPGS-F, the following graphic elements may be generated:

- straight lines
- set of straight lines
- circles/circle areas
- text
- markers
- functions

All graphic elements may be hardware generated by the graphic device.

The elements are defined in a user defined coordinate system (2-or 3-D Kartesian) and are fully transformable.

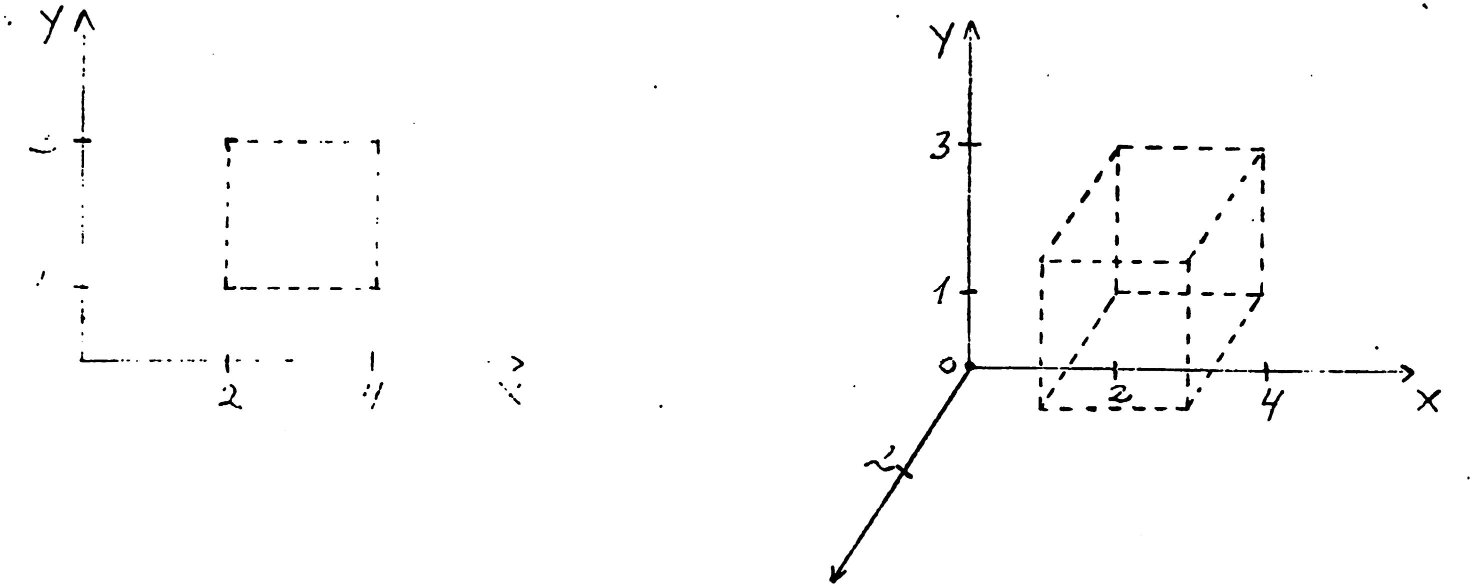
4. COORDINATE SYSTEMS

The GPGS-F user defines the graphic elements related to a userdefined coordinate system (2- or 3-D kartesian).

For the graphic device's drawing surface, a standardized coordinate system (2- or 3-D kartesion) is used. In this system the length of the shortest side of the device is set to be 1.0.

When the user defines a drawing in user coordinates, the system will map this drawing into the device's coordinates. The user defines the part of the user coordinate system which is to be mapped into the device's drawing surface by specifying a window. Examples of 2D and 3D windows are shown in fig. 2.

The part of the device's surface where the data inside the window are to be shown is defined by setting a viewport. Example of viewport and mapping from user coordinates into device coordinates are shown in fig. 3.



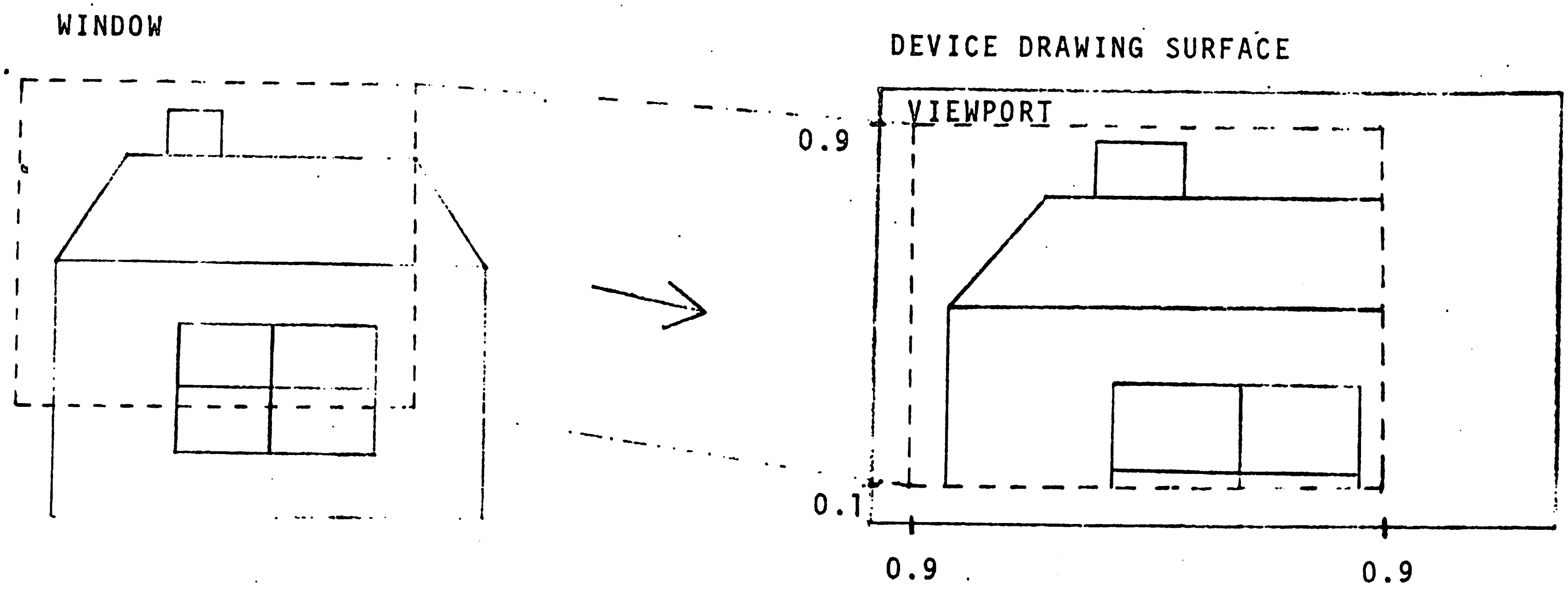
2-D WINDOW:

```
DIMENSION W (4)
DATA W / 2.0, 4.0, 1.0, 3.0/
CALL WINDW (W)
```

3-D WINDOW:

```
DIMENSION W(6)
DATA W/ 2.0, 4.0, 1.0, 3.0, 0.0, 2.0/
CALL WINDW3 (W)
```

FIG. 2 EXAMPLES OF 2D AND 3D WINDOWS


WINDOW
DEVICE DRAWING SURFACE
VIEWPORT
0.9
0.1
0.9
0.9

Defined picture in
user's coordinates

Definition of viewport:

```
DIMENSION V(4)
DATA V / 0.1, 0.9, 0.1, 0.9 /

CALL VPORT (V)
```

Fig. 3 VIEWPORT AND MAPPING FROM USER TO DEVICE COORDINATES

When a 3-D picture is shown on a 2-D drawing surface a projection is made. By default the picture will be projected parallelly into the X/Y plane part of the window box and then mapped into the viewport. Later in this paper other projections will be discussed.

5. TRANSFORMATIONS

GPGS-F system transformations change the coordinates a user passes to the system before they are viewed through the window. The change is carried out as a matrix multiplication of the input coordinate elements and a transformation matrix. Let (x,y,z) be the input points and T the current transformation matrix (homogenous), then the transformed point (x', y', z') will be:

$$[x', y', z', 1] = T \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

As the GPGS-F user can access the transformation matrix, any type of transformation that can be contained within the 4 x 4 square matrix can be performed.
Some transformation types are standard, and special routines provides automatic modification of the transformation matrix.

A list of the available transformations without user manipulation is given below:

- translation
- scaling
- rotation
- shearing

The effect of these transformation types are shown in fig. 4 based on examples in 1 .

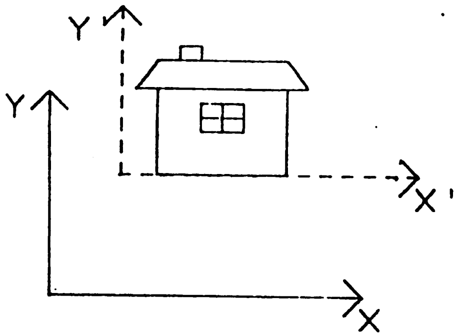
TRANSFORMATIONS

XLAT(Xdisp,Ydisp)

XLAT3(Xdisp,Ydisp,Zdisp)

Example:

CALL XLAT(20.0,40.0)
CALL HOUSE



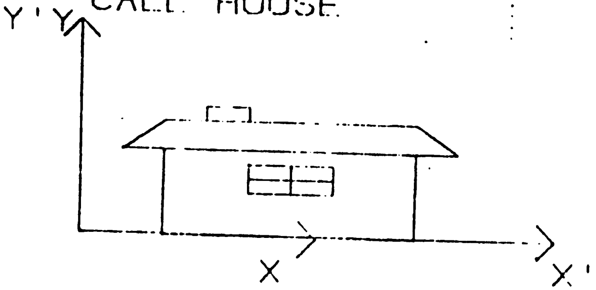
SCALE

SCAL(Scale,Iaxis)

Iaxis=0 X,Y,Z axis
=1 X axis
=2 Y axis
=3 Z axis

Example:

CALL SCAL(2.0,1)
CALL HOUSE



ROTATE

ROTA(Angle,Iaxis)

ROTAD(Dangle,Iaxis)

Angle=radians
Dangle=degrees

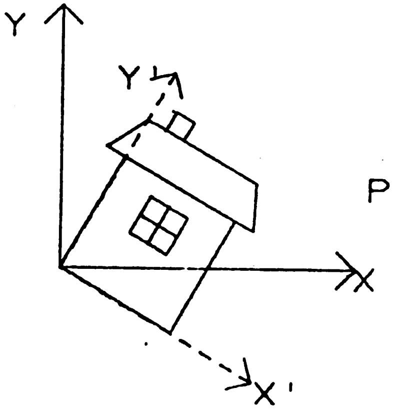
Iaxis=1 X-axis
Iaxis=2 Y-axis
Iaxis=3 Z-axis

Example:

CALL ROTA(-0.524,3)
CALL HOUSE

or

CALL ROTAD(-30.0,3)
CALL HOUSE



Positive direction of rotation

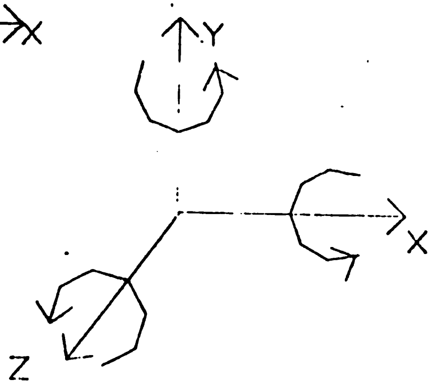
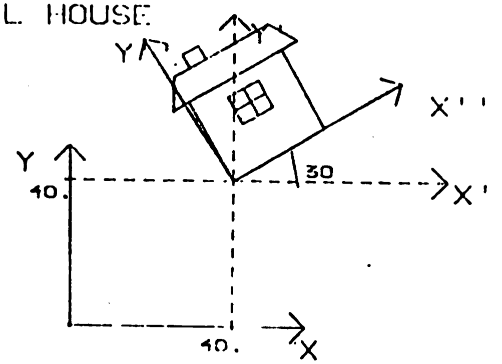


FIG. 4. EXAMPLE OF BASIC TRANSFORMATIONS

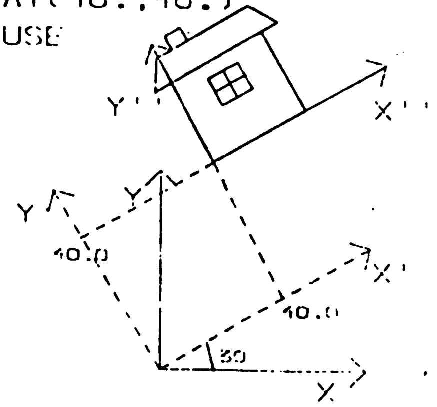
Combining Basic Transformations

```
CALL XLAT(40.,40.)
CALL ROTAD(30.,3)
CALL HOUSE
```



Y''
X''
Y
40.
30
X'
40.
X

```
CALL ROTAD(30.,3)
CALL XLAT(40.,40.)
CALL HOUSE
```



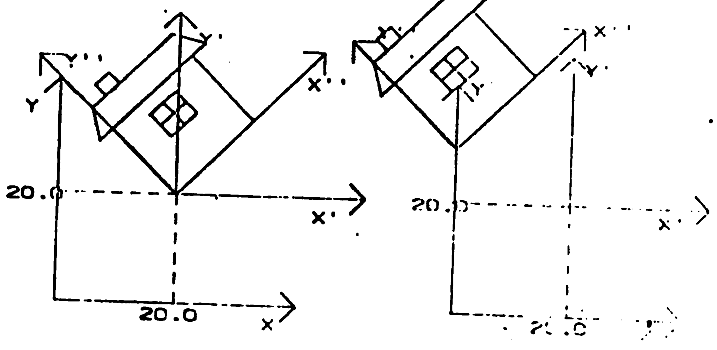
Y'
X''
Y
Y'
40.0
X'
40.0
30
X

PICTURE MODE - SPACE MODE

```
CALL MODTRN(I)
 I=0 activate space mode
                    =(default)
 I=1 activate picture mode
```

```
CALL XLAT(20.,20.)
CALL ROTAD(15.,3)
CALL HOUSE
```

SPACE MODE　　PICTURE MODE



Y''
Y'
X''
Y
20.0
X'
20.0
X
X''
Y'
20.0
X'
20.0
X

Transformations relative:

to current coordinate system　　to original coordinate system

FIG. 5 COMBINING TRANSFORMATIONS

Transformations may be combined, and are relative to either current or original coordinate system depending on the transformation mode. (See fig. 5, from [2])

The current transformation may be stacked in up to 10 levels for later restore. This is useful for example in a drawing subroutine with internal transformations.

6. PROJECTIONS

When a 3-D drawing is viewed on a 2-D device, the drawing will be projected parallelly into the X-Y plane if nothing else is specified.

Routines for spesification of the projection are provided by GPGS-F.

Two types of projections are provided, parallell and perspective. The projection is set up by means of a single routine call specifying the eye point. Examples of projections are shown in appendix A.

7. PICTURE SEGMENTS

Picture segments are a collection of graphic elements forming a picture or picture part that may be separately identified for example for lightpen picking.

Picture segments may be stored in primary or on secondary storage by means of a pseudo driver for later use.

Picture segments fetched from such store may be retransformed. Figure 6 shows how this facility is linked to GPGS-F.

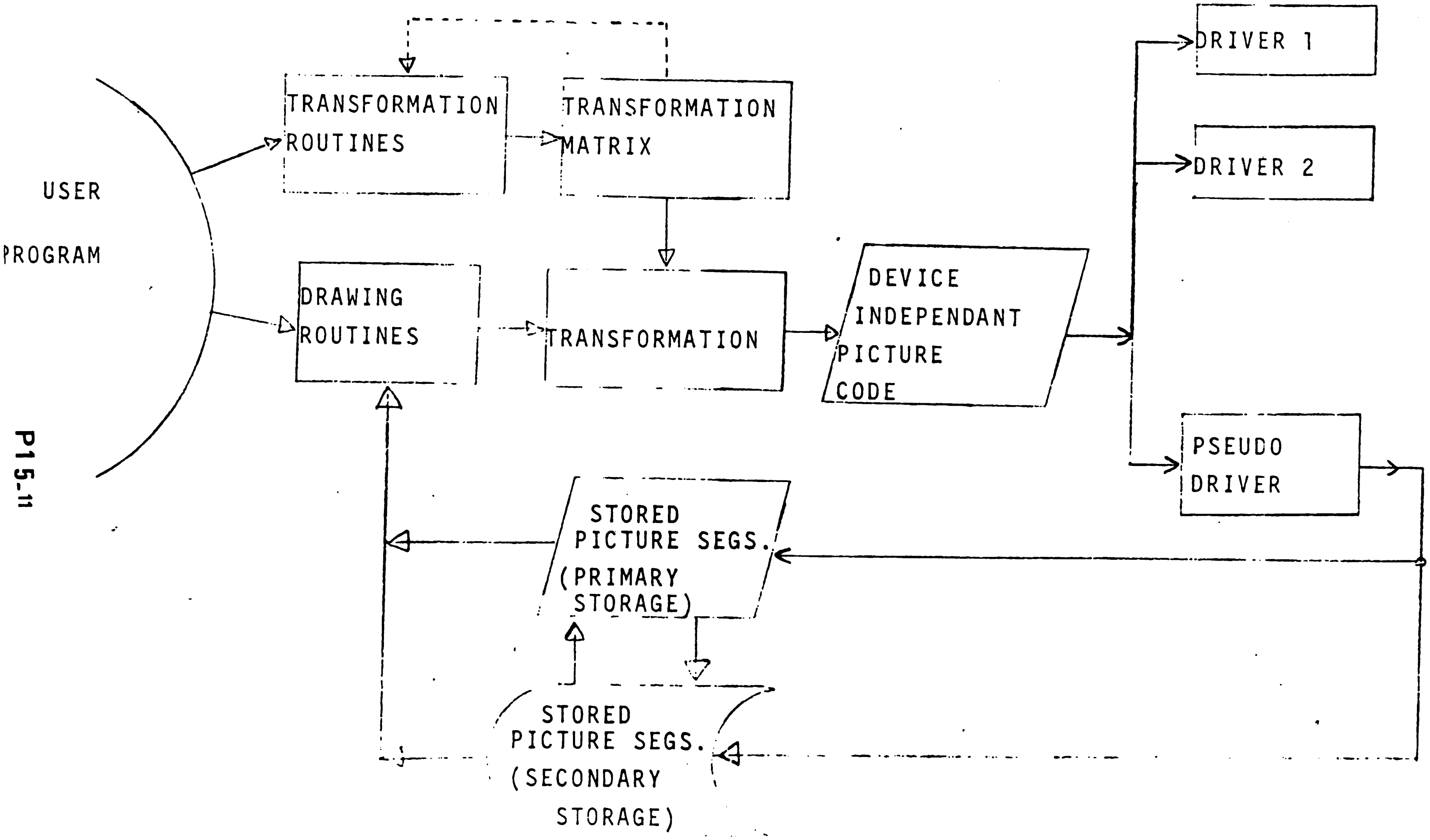
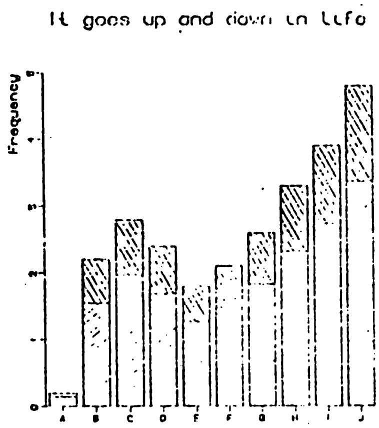

USER
PROGRAM
TRANSFORMATION ROUTINES
TRANSFORMATION MATRIX
DRAWING ROUTINES
TRANSFORMATION
DEVICE INDEPENDANT PICTURE CODE
DRIVER 1
DRIVER 2
PSEUDO DRIVER
STORED PICTURE SEGS. (PRIMARY STORAGE)
STORED PICTURE SEGS. (SECONDARY STORAGE)

FIG. 6 STRUCTURE FOR STORING OF PICTURE SEGMENTS

8. INTERACTIONS

For access of interactive facilities of the graphic devices GPGS-F uses reference to logical tools. In this way device independany is ensured.

An example showing how this works is that digitizing equipment for an interactive plotter, crosshair for a storage tube and graphic curser for a raster graphics terminal will all be considered as the same tool from GPGS-F.



9. GRAPH AND HISTOGRAM PLOTTING

Also available is a subroutine package for general graph and histogram plotting called GRAPHISTO 3 . Output of computed results is an often used task and having the results plotted gives a much better view of the data than having them printed in columns on a printer.

This package is designed to remove the programmer from doing all the drawing details like finding range of datas, drawing axes et.

It uses GPGS-F basic routines for drawing and will therefore give the user easy access to all the graphic output devices available through GPGS-F. Other facilities like changing pen colour to get different colours on different curves, plotting in true scale, and so on can be used together with GRAPHISTO.

GRAPHISTO provides 4 so-called 'chart' routines that will in answer to a single call draw a complete diagram with annotated axis, texts on axis and datapoints.

The types of plots provided through these 4 routines are:

- Histogram with labels under each bar and linear or logaritmic axis in x and y.
- Table of lines with straight lines between plots.
- Smooth curve through specified points.
- Pie chart.

These 4 routines use the basic GRAPHISTO routines as axis drawing, range computation, 'nice' value computation and curve plotting. The lower level routines are also available to the user and offers possibilities for sophisticated non-standard plots like multiple axis, marking special data points etc. Appended to this chapter are some plots to demonstrate the use and possibilities of GRAPHISTO.

Available facilities are:

Chart plotting:

- Simple and smooth (cubic spline) curves in different linestyles.
- Histograms. They may be plotted with or without hatching of bars in any angle.
- Pie charts with texts and percentage of total pie.

Axis drawing

- Near to plot on either side of plot.
- Through any data or page value.
- Several parallell with different units.

Axis annotation:

- On either side of axis
- Numeric or text labels
- Any character size and angle
- Upper and lower case
- Extra tick marks
- Title

Grid:

- Along x- or y-axis
- Linear or logaritimic
- Any line type or '+' at grid crossings

Dataplotting:

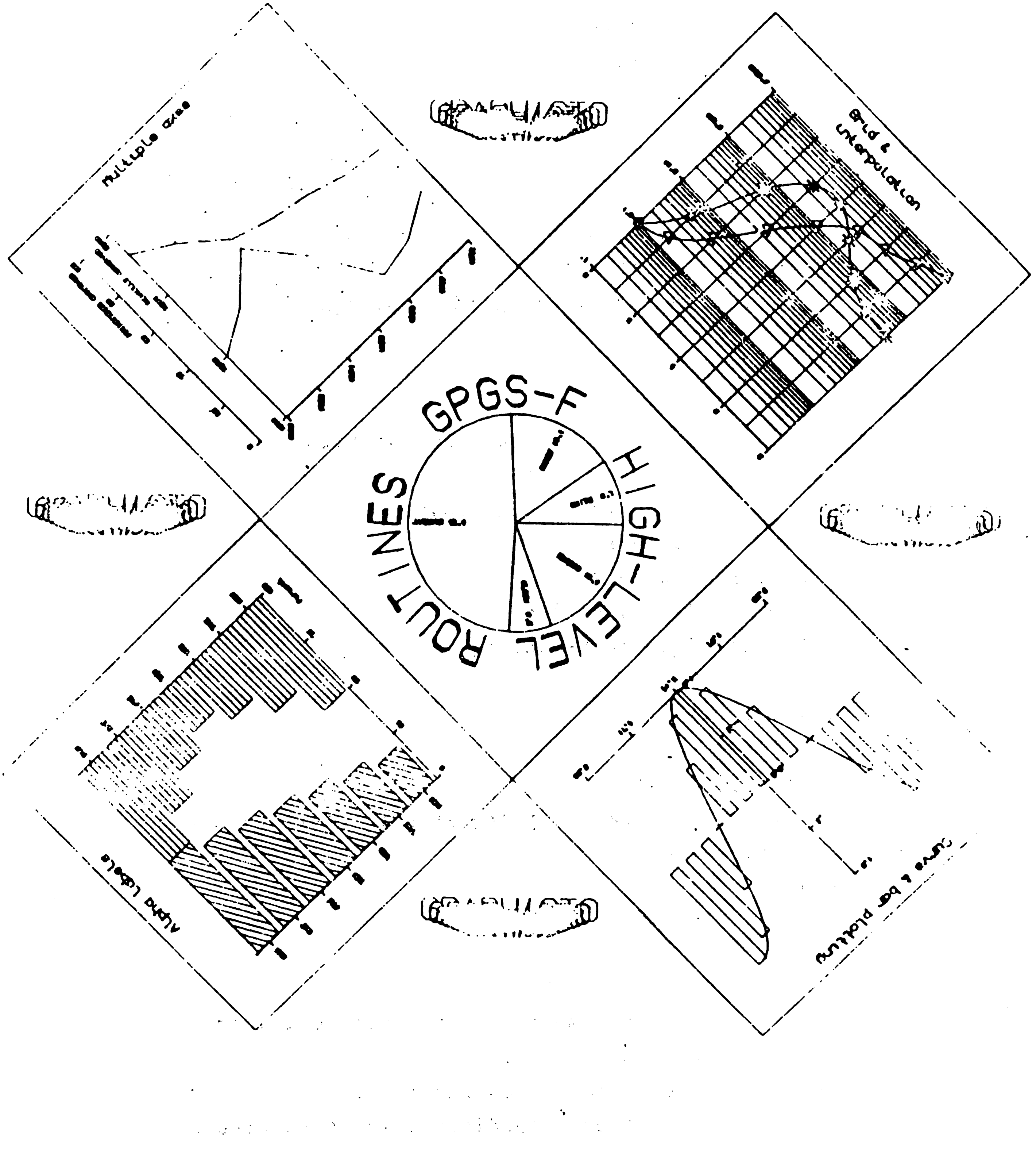
- Table of values in x- and y or functions
- Simple connected points
- Interpolated curve through points
- Markers at points
- 'Undefined' points
- Automatic data indexing
- Automatic data incrementing

Page layout:

- Centered heading
- Positioning of dataplotting area in users window
- Bar and curve legends
- Frame

Miscellaneous:

- 'Best-fit' range computation of data contained in array or function.
- 'Best-fit' label format computation.
- Conversation between coordinates in users window and in plotting coordinates.


GPGS-F HIGH-LEVEL ROUTINES
Multiple axes
Grid & interpolation
Alpha labels
Curve & bar plotting

Error Reporting:

- The routines in the GRAPHISTO system uses the error system of GPGS-F. This includes parameter checking and print of routine number, where error occured. Also the number of calls made to GRAPHISTO routines and wrong parameters are printed. This makes it easier for the user to find place and reason of error.

10. 3-DIMENSIONAL SURFACE PLOTTING

Newly developed by the graphic group at RUNIT is a package called SURRENDER 4 (SURface RENDERing) that is presented to the user as a means for visualizing bivariate surfaces on some graphic device. This package uses GPGS-F basic routines for drawing and GRAPHISTO routines for axes drawing and curve smoothing.

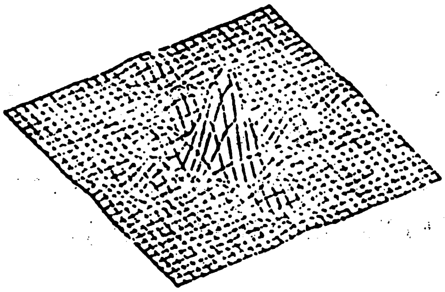
Base for all SURRENDER plotting is a rectangular x-y grid (matrix) with z-values in each node. A surface with M grid points in x-direction and N grid in y-direction will be stored in a Fortran DIMENSION ARRAY (N,M).

This grid may be rendered as a 3-D perspective plot of the grid (Isolines for x and y) with hidden lines removed or as a 2-D contour map (Isolines for z). Also other usefull facilities like drawing axes, marking points etc. are available and will be further explained later.

The package is built up much the same way as GRAPHISTO with some routines that makes a complete plot in one call, and others to add features for a more sophisticated plot.

Example of minimum effort plot:

```
      DIMENSION IWORK(200),VP(4),ZMAT(31,31),IOPT(3)
      DATA IDEV/8/,VP/0.0,0.3,0.0,0.3/
      DATA IBLUE/20/,IRED/60/
C
C  COMPUTE THE FUNCTION 'SIN(X)*SIN(Y)/(X*Y)'
C
      DO 1000 IY=-15,15
         Y=FLOAT(IY)
         SINYY=1.0
      IF (IY.NE.0) SINYY=SIN(Y)/Y
      DO 1000 IX=-15,15
         X=FLOAT(IX)
         SINXX=1.0
      IF (IX.NE.0) SINXX=SIN(X)/X
         ZMAT(IX+16,IY+16)=10.0*SINXX*SINYY
 1000 CONTINUE
C
C  MAKE MINIMUM EFFORT PLOT
C
         CALL NITDEV(IDEV)
         CALL BGNPIC(1)
         CALL PLOMA3(ZMAT,31,31,-15.,15.,-15.,15.,IWORK,200)
         CALL ENDPIC
```

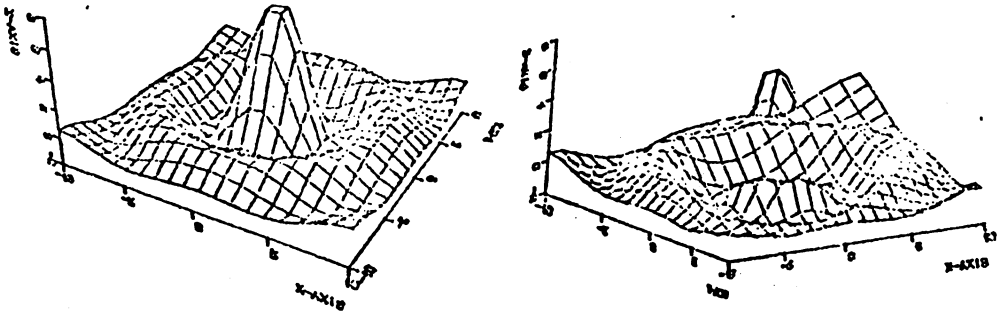


By default the hidden lines will be removed. To do this the system uses a working array to be supplied by user. As the needed size of this array is dependant upon the number of points to plot, this method gives no restrictions about number of points.

Setting focal point and eye position. The surface may be seen from any point in space and some routines are used to set this point either using cartesian or spherical coordinate system. The viewing may be either axonometric or perspective.

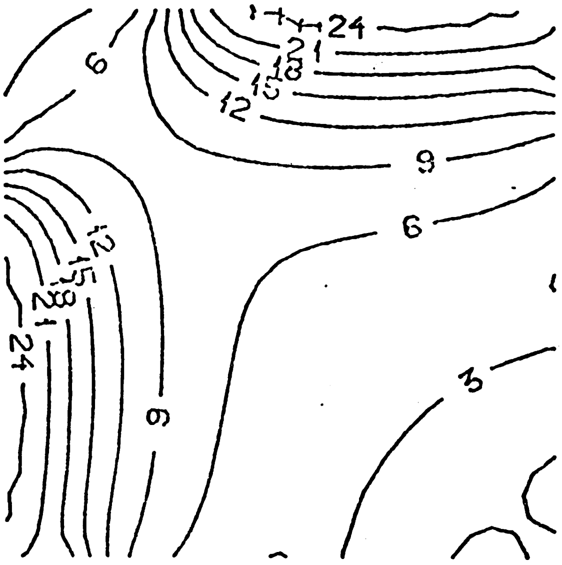
Adding axes to the plot.

Axes may be added by a call to a single routine giving standard axis annotation or by several calls to GRAPHISTO Axes routines for special labels and format.



Contour plots.

Contour plots consists of isolines in the z-direction. The range and number of contour lines may be given, and they may be added to the perspective plot or plotted as a separate 2-dimensional plot.



Default contour plot.

The SURRENDER package is not yet available on HP-3000, but this can easily be done on user requests.

11. USING GPGS-F ON HP-3000.

The HP-3000 version of GPGS-F is kept as near up to the standard version as possible and only a few routines has been changed from the users point of view. As seen from the foregoing text the system contains of many logical parts like the basic routines, a device driver, a plotting package and so on. 2 USL-files are used to contain the system and these are segmented into a number of segments each containing a logical part of GPGS-F.

One USL-file contains the basic routines and the most common used drivers. This is copied to users own USL with FCOPY.

The other contains the rest of GPGS-F, and wanted segments are copied using the SEGMENTER. This is done to save disc space for the user and preparation time.

A RL-file must be referenced at preparation time. This file contains dymmy entry-points for drivers and buffer-filesystem when not in use.

A complete set up for running GPGS-F on HP-3000:

```
: COMMENT copy basic part of GPGS-F
: RUN FCOPY.PUB.SYS
  FROM = GPGS.GPGS; TO = USERUSL; NEW EXIT
: COMMENT compile your program
: FORTRAN USERSOOR, USERUSL
: PREP USERUSL, USERPROG; RL = GPGSRL.GPGS
: RUN USERPROG
```

Aproximate segment sizes are as follows (decimal)

Basic drawing routines:	8400 (6 segments)
Incore buffer system:	2000
GRAPHISTO	9100 (3 segments)

Drivers:

Pseudo and file	400 each
Tektronix screens	2500 each
" plotter 4662	3100
Hewlet Packard 7221	3900
" 2648	3500

12. ERROR HANDELING

Inproper or inconsistent use of GPGS-F is possible and requires informative error messages.

GPGS-F provides the following informations in an error message:

1.	Error number	- indicates the type of error which occured
2.	GPGS subroutine no.	- identifics the routine that was called causing the error
3.	Parameter no.	- indicates which parameter, if any, caused the error
4.	Severity code	- indicates which action GPGS-F will take
5.	Call no	- indicates number of calls to GPGS since the program started
6.	Argument value	- value of the parameter in error, if any.

A GPGS-F user can instead of using the standard error handeling code his own error routine and in this way prevent error termination. This is done by dynamical change of eventual arguments with bad values.

PROGRAM USED TO CREATE FIGURES ON TITLE PAGE

```
      PROGRAM ROTBODY
C
C     MAIN PROGRAM FOR GENERATION AND PERSPECTIVE VIEWING
C     OF A ROTATION BODY
C
      DIMENSION LTY(100),BUFF(100,2),EYE(3)
C
C     SELECT TYPE OF GRAPHIC DEVICE (ENTER DEVICE CODE)
C
      DISPLAY 'DEVICE NO.'
      ACCEPT ID
C
C     INITIATE GPGS-F WITH SELECTED DEVICE
C
      CALL NITDEV (ID)
C
C     GENERATE THE CONTOUR TO BE ROTATED CREATING THE ROTATION BODY
C
      CALL BGNPIC(1)
      CALL GENPIC(BUFF,LTY)
      CALL ENDPIC
      IPIC = 1
   10 CONTINUE
C
C     SELECT EYEPOINT FOR PERSPECTIVE VIEW
C
      DISPLAY 'EYEPOINT'
      ACCEPT EYE
C
C     STACK CURRENT TRANSFORMATION FOR LATER RESTORE
C
      CALL BGNTRN
      IF (EYE(1).GE.9999.) GO TO 100
      IF (ABS(EYE(1)+EYE(2)+EYE(3)).GT.0.01)
     1   CALL PERS (EYE(1),EYE(2),EYE(3))
C
C     CLEAR DRAWING SURFACE
C
      CALL CLRDEV(ID,0)
C
C     INITIATE NEW PICTURE SEGMENT
C
      IPIC = IPIC + 1
      CALL BGNPIC(IPIC)
C
C     DRAW HORIZONTAL CONTOUR LINES
C
      CALL HORCNT(BUFF,LTY)
C
C     DRAW VERTICAL CONTOUR LINES
C
      CALL VERCNT(BUFF,LTY)
      CALL ENDPIC
C
C     POP LATEST STACKED TRANSFORMATION
C
      CALL ENDTRN
      GO TO 10
  100 CONTINUE
C
C     END  -  TERMINATE GPGS-F
C
      CALL RLSDEV(ID)
      STOP
      END
```

```
C
C
      SUBROUTINE GENPIC (BUFF,LTY)
C
C    SUBROUTINE TO GENERATE A CONTOUR CONSISTING OF
C    UP TO 99 STRAIGHT LINES (VISIBLE OR INVISIBLE)
C
      DIMENSION LTY(100),BUFF(100,2)
      DATA IL/2HL /,IP/2HP /,IS/2HS /
C
C    THE FIRST INPUT POINT IS USED AS A POINT ON THE VERTICAL
C    ROTATION AXIS
C
      CALL CURWIN (ICH,DX,DY)
C
C    THE COORDINATE SYSTEM IS TRANSLATED TO ENABLE DRAWING
C    RELATIVE TO AN ORIGIN IN THE FIRST POINT ON THE CONTOUR
C
      CALL XLAT (DX,DY)
C
C    MARK FIRST POINT WITH A CROSS (MARKER NO 4)
C
      CALL LINE (0.,0.,0)
      CALL MARKER (4)
      BUFF(1,1) = 0.
      BUFF(1,2) = 0.
      LTY(1) = 0
C
C    GENERATE THE OTHER POINTS ON THE CONTOUR,
C      END OF CONTOUR IS INDICATED BY ENTERING 'S' WITH THE POINT,
C       BLANK LINE TO POINT WITH 'P'
C
      DO 10 I = 2,100
      CALL CURWIN (ICH,X,Y)
      IF (ICH.EQ.IS) GO TO 20
      LINTYP = 1
      IF (ICH.EQ.IP) LINTYP = 0
      X = X - DX
      Y = Y - DY
C
C    VERIFY INPUT BY DRAWING
C
      CALL LINE (X,Y,LINTYP)
      BUFF(I,1) = X
      BUFF(I,2) = Y
      LTY(I) = LINTYP
   10 CONTINUE
      RETURN
   20 CONTINUE
C
C    LINETYPE -1 INDICATES END OF DATA ON CONTOUR
C
      LTY(I) = -1
      RETURN
      END %
      SUBROUTINE CURWIN (ICHAR,XPOS,YPOS)
C
C     CURWIN RETURNS A COORDINATE POINT AND A CHARACTER
C     INPUT BY MEANS OF THE GRAPHIC DEVICE'S POSITION INPUT TOOL.
C
C     PARAMETERS:
C         OUTPUT: ICHAR - TYPED CHAR (A1 FORM.)
C                 XPOS  - X COORDINATE OF INPUT POINT
C                 YPOS  - Y COORDINATE OF INPUT POINT
C
      DIMENSION IED(2),FAR(2)
      DIMENSION W(6),V(6)
      DATA IED /201,-1/
C
C     FETCH CURRENT DEVICE NO., WINDOW AND VIEWPORT FROM GFGS-F
C
      CALL DATDNO (ID)
      CALL DATWIN (W)
      CALL DATVP (V)
C
C     SET UP POSITIONING TOOL FOR ACTIVE DEVICE
C     AND WAIT FOR INPUT FROM TERMINAL USER
C
      I = INWAIT (-1.,IED,ICHAR,1,FAR,2)
C
C     CONVERT FROM DEVICE TO USER COORDINATES
C
      YPOS = W(1) + FAR(1)*(W(2)-W(1))/(V(2)-V(1))
      YPOS = W(3) + FAR(2)*(W(4)-W(3))/(V(4)-V(3))
      IF (ID.NE.11.AND.ID.NE.12) RETURN
C
C     IF DEVICE IS INTERACTIVE PLOTTER, NO CHARACTER CAN BE ENTERED
C     FROM THE PLOTTER DIRECTLY  -
C     CHARACTER IS THE READ FROM THE ATTACED TERMINAL.
C
      CALL WRITOL (2,0,0,0.,0)
    1 FORMAT (" ENTER CHARACTER:")
      WRITE (6,1)
    2 FORMAT (A1)
      READ (5,2) ICHAR
      RETURN
      END
```

```
C
C
      SUBROUTINE VERCNT (BUFF,LTY)
C
C    SUBROUTINE TO DRAW THE VERTICAL CONTOUR
C    IN 35 POSITIONS ROTATED AROUND THE ROTATION AXIS
C
      DIMENSION LTY(100),BUFF(100,2)
C
C    STACK CURRENT TRANSFORMATION
C
      CALL BGNTRN
      DO 100 J = 1,35
C
C    DRAW CONTOUR IN CURRENT COORDINATE SYSTEM
C
      DO 10 I = 1,100
      LINTYP = LTY(I)
      IF (LINTYP.LT.0) GO TO 20
      CALL LINE (BUFF(I,1),BUFF(I,2),LINTYP)
   10 CONTINUE
   20 CONTINUE
C
C    ROTATE COORDINATE SYSTEM 10 DEGS. AROUND Y-AXIS (ROT. AXIS)
C
      CALL ROTAD (10.,2)
  100 CONTINUE
C
C    POP LATEST STACKED TRANSFORMATION
C
      CALL ENDTRN
      RETURN
      END

C
C
      SUBROUTINE HORCNT (BUFF,LTY)
C
C    SUBROUTINE TO DRAW HORIZONTAL CONTOUR LINES (CIRCLES)
C
      DIMENSION LTY(100),BUFF(100,2)
C
C    LOOP OVER ALL POINTS ON THE VERTICAL CONTOUR
C
      DO 100 I = 1,100
      IF (LTY(I).LT.0) RETURN
      X = BUFF(I,1)
      Y = BUFF(I,2)
      IF (ABS(X).LT.0.001) GO TO 100
C
C    DRAW HORIZONTAL CIRCLES THROUGH A POINT ON THE VERTICAL
C    CONTOUR LINE WITH CENTRE IN THE ROTATION AXIS
C
      CALL LINE (X,Y,0)
      CALL CIRD3 (0.,Y,0.,X,Y,X,360.,1)
  100 CONTINUE
      RETURN
      END
```

REFERENCES.

1) M. Zachrisen : GPGS-F User's Guide, 3rd edition TAPIR Forlag 1978.

2) M. Zachrisen : GPGS-F Cource Manual, RUNIT 1977.

3) M. Zachrisen : GRAPHISTO, a subroutine package for general graph and histogram plotting with GPGS-F.
RUNIT report STF14 A78019.

4) M. Zachrisen : PRELIMINARY description of SURRENDER - a subroutine packet for rendering bivariate surfaces.

CENTRALIZED DATA PROCESSING: AN APPROACH TO SYSTEMS DEVELOPMENT

BY
JAMES G. OLIVERIO
L. DAVID SALYER
GTE DATA SERVICES INCORPORATED

BACKGROUND

GTE Data Services Incorporated is a wholly-owned subsidiary of General Telephone & Electronics Corporation (GTE). The Company provides data processing services to GTE telephone operating subsidiaries and to independent telephone companies throughout the country. The 19 GTE domestic and international telephone companies employ some 116,000 persons and serve more than 16.6 million telephones.

Prior to the formation of GTE Data Services, GTE realized that increased demands for information within the GTE telephone companies were causing data processing costs to soar. Even short-range forecasts by the telephone companies indicated dramatic rises in data processing personnel and computer rental costs to meet their ever-increasing needs.

Consequently, GTE conducted studies to find an economical approach toward development and usage of data processing that could meet the challenges of the future. As a result of these studies, GTE Data Services was formed to:

1) Provide computer systems, programming and processing services to the GTE telephone companies.

2) Develop and implement a series of central, computer based information systems to provide all GTE telephone companies with timely data on a broad range of specialized subjects. This information enables the companies to operate more efficiently and greatly aids their management in decision-making and planning.

3) Provide the telephone industry, through our time-sharing library, a highly reliable means of solving long range and daily problems in every area of operation including finance, budgeting, marketing, engineering and revenue requirements.

ORGANIZATION

GTE Data Services employs more than 2,000 persons throughout its 12 data centers. These 12 data centers are divided into three regions, designed to efficiently meet the processing requirements of the three GTE telephone company regions.

The western region includes data centers in Los Angeles, California and Everett, Washington. The northern region contains data centers in Fort Wayne, Indiana; Bloomington, Illinois; Muskegon, Michigan; Johnstown, New York; Marion, Ohio; Erie, Pennsylvania; and Sun Prairie, Wisconsin. The southern region includes data centers in San Angelo, Texas; Durham, North Carolina; and Tampa, Florida.

MAJOR ACHIEVEMENTS

GTE Data Services was organized primarily to meet several challenges facing the data processing operations of the GTE telephone companies. These include: slowing the rapid rise of data processing costs, reducing the number and variety of computer systems in use, developing generalized computer programs for use by all telephone companies, and handling the tremendous volume of information.

The Company met these challenges head on. Costs were reduced by purchasing computer hardware, rather than leasing or renting it; by using fewer but more powerful computers; by eliminating duplication of programming efforts; and by centralizing the purchasing and control of equipment.

The Company has formed regional data centers in each part of the country, where large-scale computers are available to handle the needs of one large telephone company or several smaller companies.

Regional and divisional data centers using standardized computers and procedures, coupled with centralized systems and programming development, enable GTE Data Services to provide its customers with efficient and timely services.

SYSTEMS DEVELOPMENT

Our centralized Telephone Systems Group, comprised of some 400 data processing and telephone professionals, plays a leading role in our continuing challenge to meet the growing needs of the GTE telephone companies. Aside from developing major data processing systems for the GTE companies, an increasingly important goal of this staff is the development and implementation of a Business Information System (BIS) to assist management with the control and decision-making process while minimizing data processing costs and optimizing data handling.

BIS represents a long range concept of development, with its various systems designed to fit together to provide a comprehensive data base for telephone company management.

In addition to BIS, the development of Special Products, Interim Systems, and Time Sharing are three other key items for which we are responsible. Special Products are administrative computer systems which study equipment usage and facilities administration by interfacing the output of electronic central office switching equipment to existing management or business systems, such as customer billing.

When the developmental time factor is critical, Interim Systems are developed in a brief time frame using standard computer languages. These systems stand alone and are not necessarily dependent on BIS systems or information.

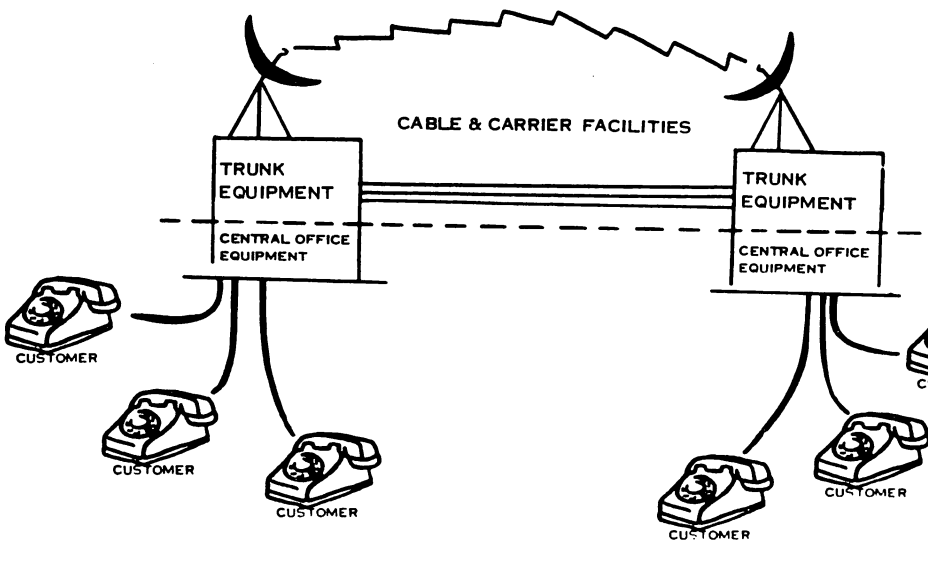
When processing response time is critical or there is a need for the user to interact with the computer in a problem-solving mode of operation, systems are developed by the Time Sharing group.

THE APPLICATION

The application that we will be discussing deals with the record keeping associated with the design, selection of the equipment and facilities (or assignment), and maintenance of Telephone Trunk networks. Customer telephones are served through individual Central Offices. When a customer calls someone connected to a different central office than his own, his call is carried via the trunk network to the receiving central office and then to the phone being called. Whenever you make a long distance telephone call or local call that must go through more than one central office, you are using a circuit that is commonly called a "Message Trunk". This "Message Trunk" is composed of items of equipment in the central offices and cable pairs or carrier channels (facilities) which go between central offices. Also, if you have any kind of telephone service other than your ordinary home or business phone, you probably have what is referred to as a "Special Service Circuit". Examples of Special Service Circuits are WATS service, private line data circuits, or direct voice circuits which bypass the telephone switching mechanisms. As with "Message Trunks", these special service circuits must use equipment and facilities to perform their prescribed function.

EXHIBIT I

TRUNK NETWORK


CABLE & CARRIER FACILITIES
TRUNK EQUIPMENT
CENTRAL OFFICE EQUIPMENT
TRUNK EQUIPMENT
CENTRAL OFFICE EQUIPMENT
CUSTOMER
CUSTOMER
CUSTOMER
CUSTOMER
CUSTOMER
CUSTOMER

The following is an oversimplification of how these records are maintained manually.

1) When a circuit is to be installed, an engineer will design the circuit on a paper, specifying the central offices through which the circuit is to go and the type of equipment to be used at each central office.

2) The engineer will then forward this design to an assignment clerk who maintains work books on the facilities and equipment installed throughout the telephone company. The clerk will go through these books to find available facilities and equipment which meet the designers requirements. Once found, the clerk will write the circuit identification in the workbook next to the appropriate items, then, post the assignment information on the design form.

3) Several copies of the design form will be made and a copy sent to each location where work is to be performed in order to install the circuit as well as to various administrative locations in the company where these records are required. Often times this information will also be sent to other telephone companies which might be involved in providing the service.

4) When a circuit is removed from service, the files will have to be purged and erasures made in the work books in order to free the facilities and equipment for future use.

5) Frequently, changes will have to be made after the design forms are distributed and these changes will have to be communicated to all locations where these records are stored even though the change may not necessarily affect them.

As you can see, this creates a massive procedural problem. When you consider that an average telephone company may have 65,000 circuits, 150,000 facilities, and 200,000 items of equipment, the problem becomes acute and accuracy of the records can deteriorate to the point where they are no longer reliable.

The first step in mechanizing this application was to develop a system which would capture the manual records and point out to the users where inconsistencies existed in their records. They could then resolve these differences and return their records to a high degree of accuracy. The second step would be to provide an operational system to handle daily activity that would prevent these records from deteriorating in accuracy again.

The first step was accomplished with the Trunk Administration inventory and reconciliation system, which is a batch system that runs on an IBM 360/370 computer. This system was enhanced with a mini-computer based front end system called the Trunk Administration Record Support System (TARS). Originally written for the HP21MX, TARS has been converted to the HP3000. These systems, in tandem, provide an economical and relatively fast method of collecting, validating, and reconciling the manual records.
Five telephone companies currently are using the Trunk Administration System and three of those companies are also using TARS. Additionally, we have requests to install both TA and TARS in two more companies.

The second step of mechanization, that of providing an operational system, is primarily what we are here to talk about today. The way we established the requirements, developed the system, and implemented it on the Hewlett Packard 3000 will be discussed in some detail.

ESTABLISHING THE REQUIREMENTS

The requirements for the operational system were written by a "requirements team". The team was composed of representatives from six telephone companies, GTE Service Corporation, and GTE Data Services. This team was chartered to document the requirements for an operational system, recognizing that each company had many differing procedures and requirements. The requirements were to address those features that were common to all companies and make provisions for enabling each company to meet its own unique requirements.

The requirements team produced a "Requirements Document" (RQD) for the Circuit Network Administration System (CNAS) which was submitted to GTEDS for evaluation. GTEDS responded by proposing a modular development approach utilizing mini-computer technology and networking capabilities. In formulating its proposal, GTEDS compared the alternatives of putting this system on line to an IBM/370 or using mini-computers.

EXHIBIT II

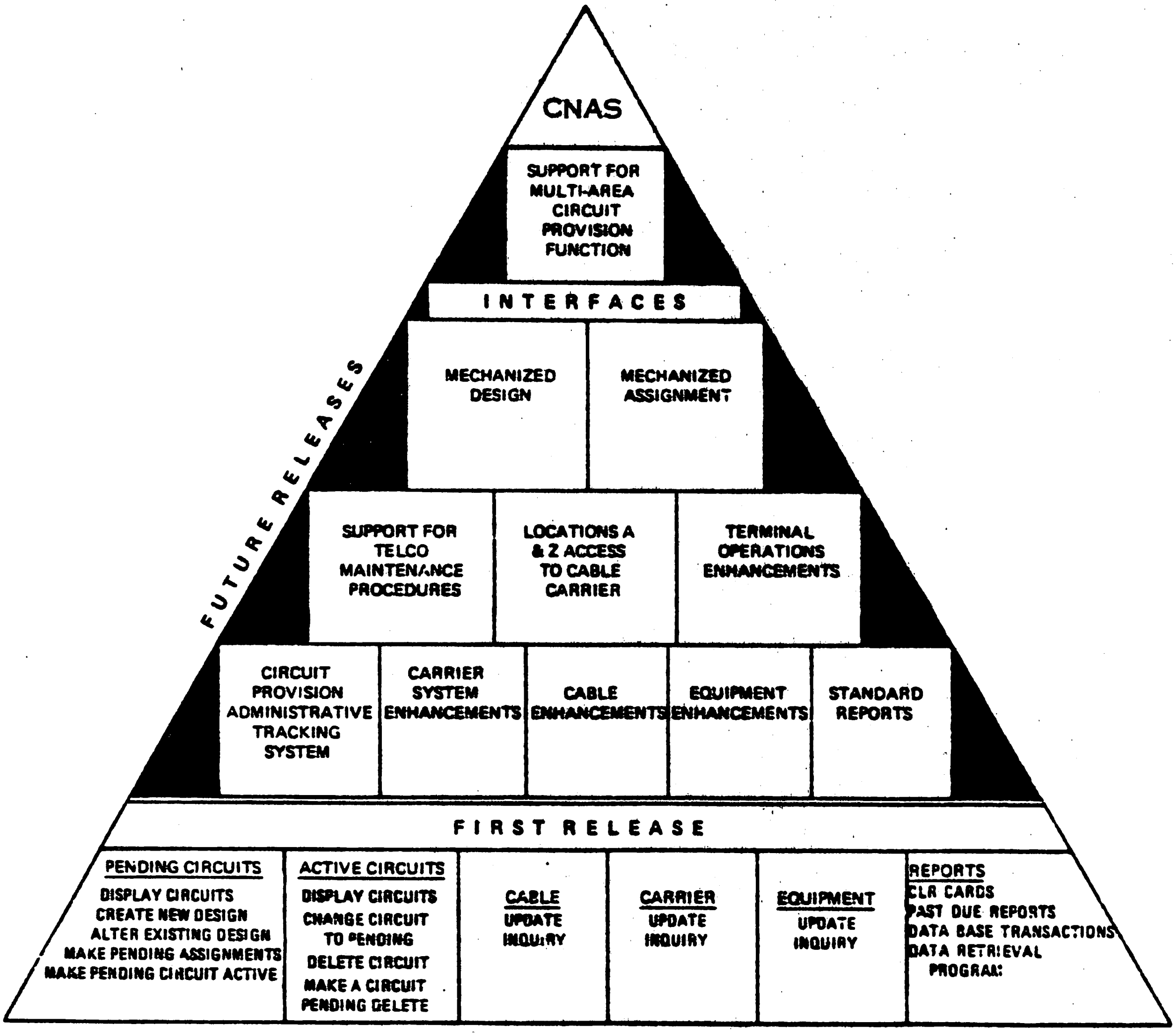

CNAS
SUPPORT FOR MULTI-AREA CIRCUIT PROVISION FUNCTION
INTERFACES
FUTURE RELEASES
MECHANIZED DESIGN
MECHANIZED ASSIGNMENT
SUPPORT FOR TELCO MAINTENANCE PROCEDURES
LOCATIONS A & Z ACCESS TO CABLE CARRIER
TERMINAL OPERATIONS ENHANCEMENTS
CIRCUIT PROVISION ADMINISTRATIVE TRACKING SYSTEM
CARRIER SYSTEM ENHANCEMENTS
CABLE ENHANCEMENTS
EQUIPMENT ENHANCEMENTS
STANDARD REPORTS
FIRST RELEASE
PENDING CIRCUITS
DISPLAY CIRCUITS
CREATE NEW DESIGN
ALTER EXISTING DESIGN
MAKE PENDING ASSIGNMENTS
MAKE PENDING CIRCUIT ACTIVE
ACTIVE CIRCUITS
DISPLAY CIRCUITS
CHANGE CIRCUIT TO PENDING
DELETE CIRCUIT
MAKE A CIRCUIT PENDING DELETE
CABLE
UPDATE
INQUIRY
CARRIER
UPDATE
INQUIRY
EQUIPMENT
UPDATE
INQUIRY
REPORTS
CLR CARDS
PAST DUE REPORTS
DATA BASE TRANSACTIONS
DATA RETRIEVAL PROGRAM

FIGURE 1 – NEW TRUNK ADMINISTRATION SYSTEM MODULAR STRUCTURE

Before doing the economic study, it was necessary to determine if the HP3000 had the capability of handling a system of this size (i.e. the largest module). To make this determination, a data base was designed, volumes established, and scenarios written for typical system activity. These scenarios were described in the form of "scripts" for the IDEA processor. We then made several runs of IDEA varying the amount of activity against the data base. The results of these runs showed that the HP3000 had the capability of handling the transaction volumes required by the system and that with DS3000 the modular approach could be implemented in a logical manner.

The economic evaluation, using five telephone companies as the basis, showed that processing the system on an IBM computer was much more expensive than processing on an HP3000 even when you considered the sharing of the IBM resources with other systems. This information was the primary basis for the GTEDS proposal.

SYSTEMS DEVELOPMENT

The proposal was accepted and authorization for development of the first module of the Circuit Network Administration System (CNAS) was given on July 29, 1977. A staff of eight Data Processing professionals was budgeted for this project. The project was organized into three functional areas headed by a Systems Supervisor;

1) A "programming" team headed by a chief programmer. The team was responsible for system design, programming, documentation, and testing of the system.

2) System Management - This area consisted of the HP3000 system manager and a general clerk. They were responsible for the HP3000 operation, data entry, and utility software requested by the programming team.

3) Conversion - Initially, this area consisted of one person and was later expanded to three. The responsibility of this area was to develop the methodology and programs to convert the data base from the Trunk Adminstration System to CNAS.

The "programming" team designed the system using structured techniques. Included in these techniques were:

1) Tops down design

2) Hierarchical Input Process and Output (HIPO) charts

3) Structured walk-thrus, and

4) The use of a librarian

During the design phase, the data entry philosophy was established, data element definitions were entered into a data dictionary system, and the data base schema was established.

Hardware requirements for each of the telephone companies were developed and it was determined that each company would require between three and six 120 mb. disc drives depending on their size, 512 kb. main memory, and a tape drive.

When the detailed design was completed, a System Specification report was prepared and distributed to the requirements team. In addition, simulated demonstrations of the system were prepared on cassettes for the HP2645 terminal. The requirements team was then called together and they were given a demonstration of the system on the terminal. By focusing a television camera on the terminal and displaying the video on a seven foot television screen, they were able to get a good understanding of how the system was designed and would work in a live environment.

As a result of this presentation, the requirements team was able to identify a number of changes or additions that they wanted in the system. It was relatively easy to incorporate the requested changes, since, up to this point, no programming had taken place.

Once the requested design changes were made, program development began in earnest. Detailed program specifications were written; again, using the "tops down" technique. Programs were oriented around user functions and thus, were modular almost as a by-product of the development method instead of by design. Program "mainlines" which incorporated several functions were developed. By using this "mainline"/"subprogram" technique for the on-line programs, maximum flexibility for system tuning purposes was maintained.

Program functions which were going to either require a significant amount of time to execute or utilize a lot of the system resources, were designed to be initiated by on-line transactions and then processed in a batch mode.

The first release in its final form contained approximately 90,000 lines of code. Of that total, 85,000 were COBOL and 5,000 were SPL. The data base schema contains 177 elements and 64 data sets. There are 67 discrete transactions in the on-line portion of the system and eight batch processes. Exhibit III on the following three pages is a representation of the data base structure that exists today.

In developing a system of this size on a mini-computer, as you might expect, many interesting problems developed. Some of the constraints of the HP that we continually had to deal with were:

Problem: Maximum Code Segment Size
Constraint: System doesn't allow for a code segment to be larger than what is specified in system generation parameter. The maximum size specified in our system is 16,384 bytes.
Resolution: Used COBOL sections. Attempted to establish uniform section sizes. Also, frequency of execution and logically related process considerations were applied in establishing the content of a section.

Problem: Maximum number of segments per program.
Constraint: System allows only 63 code segments per program. Each COBOL program generates an initialization segment.
Resolution: Established a mainline subprogram concept which allowed us to move subprograms from mainline to mainline. First consideration was to group programs with logically similar functions. Second consideration was the aggregate number of segments in the program. Where necessary, subprograms could be in more than one mainline.

Problem: COBOL initialization segment size
Constraint: If you have a large number of value clauses in your data division, the code generated for the initialization segment can exceed the maximum code segment size.
Resolution: Initialize data areas from within the programs procedure division.

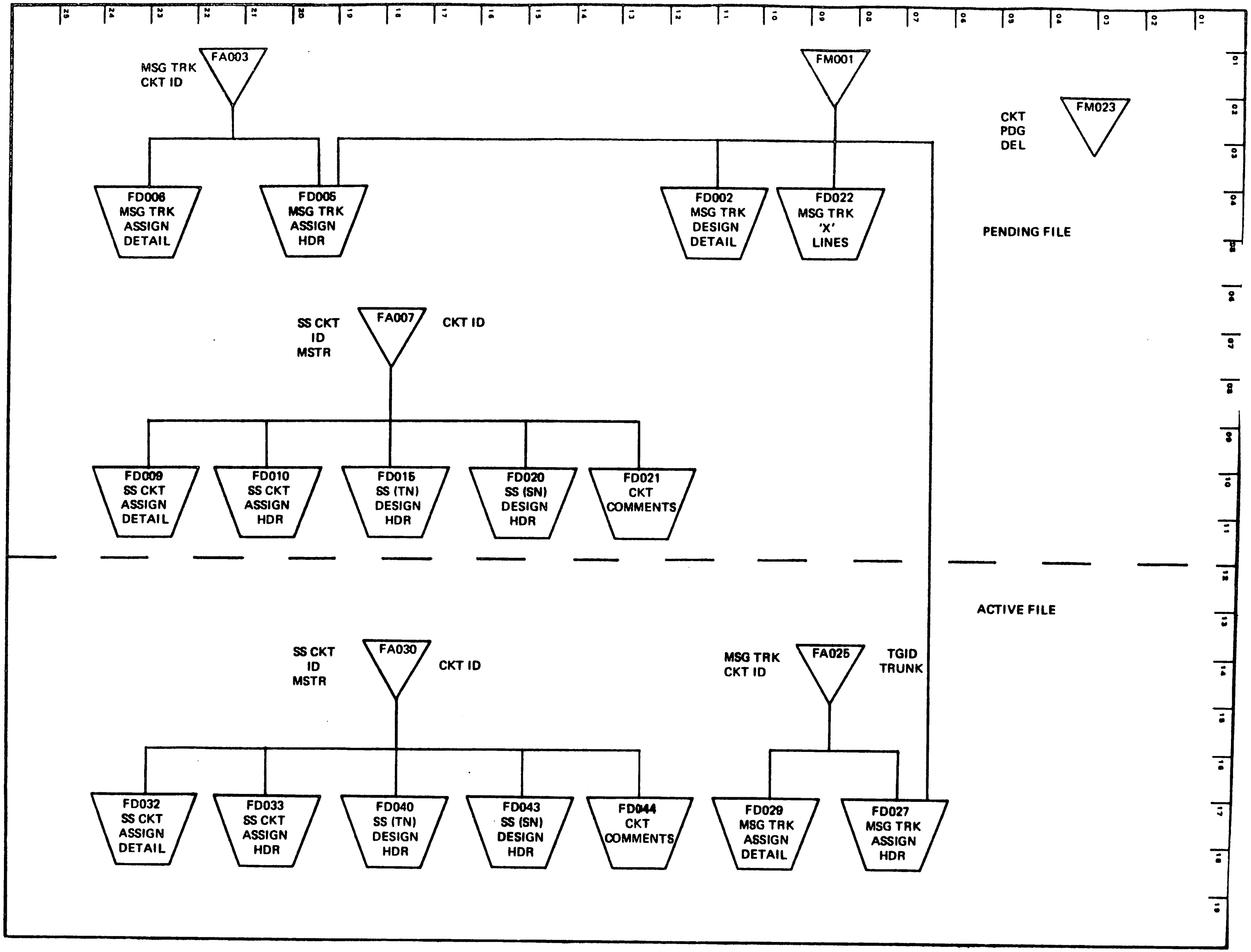

MSG TRK CKT ID
FA003
FM001
CKT PDG DEL
FM023
FD006 MSG TRK ASSIGN DETAIL
FD005 MSG TRK ASSIGN HDR
FD002 MSG TRK DESIGN DETAIL
FD022 MSG TRK 'X' LINES
PENDING FILE
SS CKT ID MSTR
FA007
CKT ID
FD009 SS CKT ASSIGN DETAIL
FD010 SS CKT ASSIGN HDR
FD015 SS (TN) DESIGN HDR
FD020 SS (SN) DESIGN HDR
FD021 CKT COMMENTS
ACTIVE FILE
SS CKT ID MSTR
FA030
CKT ID
MSG TRK CKT ID
FA025
TGID TRUNK
FD032 SS CKT ASSIGN DETAIL
FD033 SS CKT ASSIGN HDR
FD040 SS (TN) DESIGN HDR
FD043 SS (SN) DESIGN HDR
FD044 CKT COMMENTS
FD029 MSG TRK ASSIGN DETAIL
FD027 MSG TRK ASSIGN HDR

EXHIBIT III – I

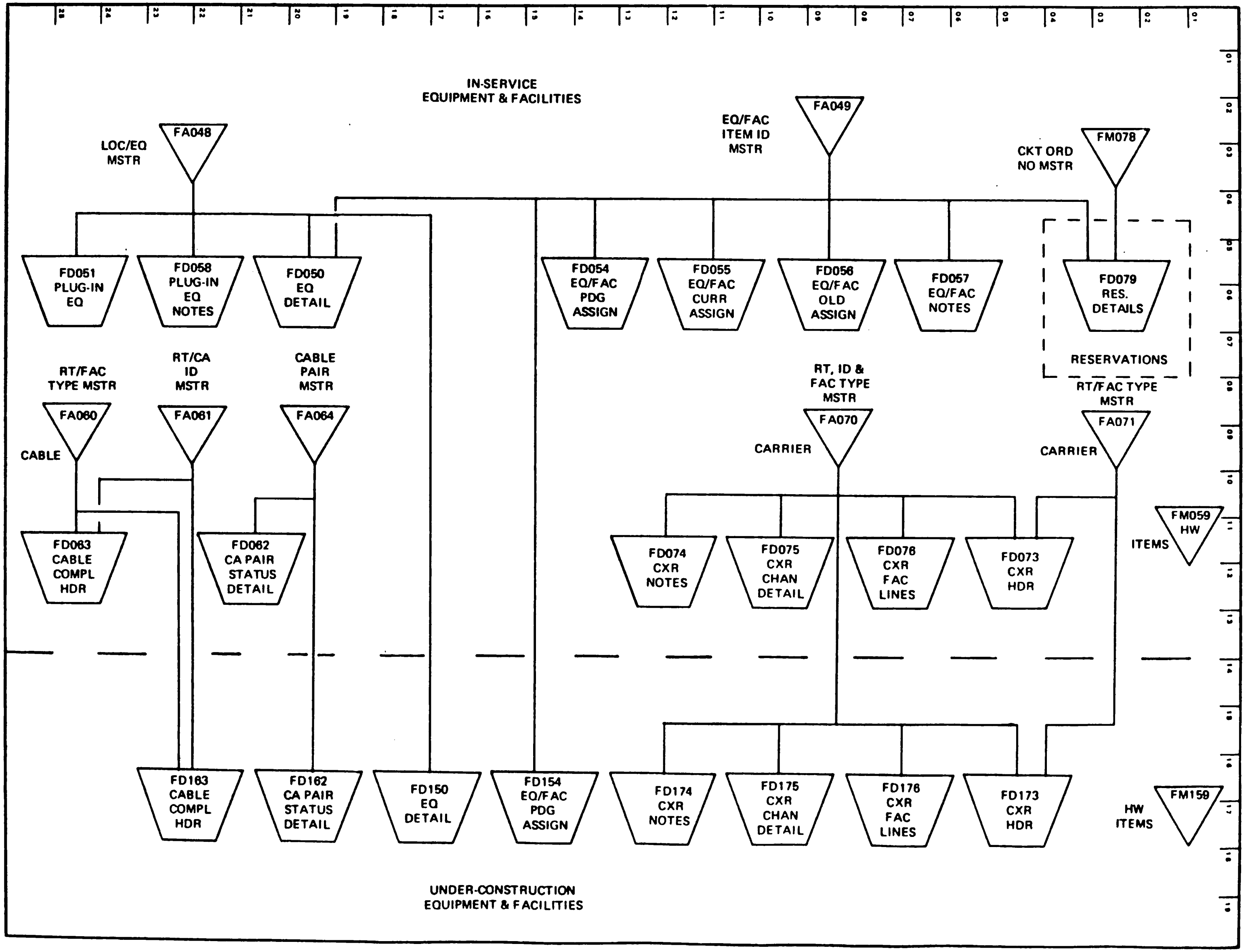

IN-SERVICE EQUIPMENT & FACILITIES
LOC/EQ MSTR
FA048
EQ/FAC ITEM ID MSTR
FA049
CKT ORD NO MSTR
FM078
FD051 PLUG-IN EQ
FD058 PLUG-IN EQ NOTES
FD050 EQ DETAIL
FD054 EQ/FAC PDG ASSIGN
FD055 EQ/FAC CURR ASSIGN
FD056 EQ/FAC OLD ASSIGN
FD057 EQ/FAC NOTES
FD079 RES. DETAILS
RESERVATIONS
RT/FAC TYPE MSTR
FA060
CABLE
RT/CA ID MSTR
FA061
CABLE PAIR MSTR
FA064
RT, ID & FAC TYPE MSTR
FA070
CARRIER
RT/FAC TYPE MSTR
FA071
CARRIER
FD063 CABLE COMPL HDR
FD062 CA PAIR STATUS DETAIL
FD074 CXR NOTES
FD075 CXR CHAN DETAIL
FD076 CXR FAC LINES
FD073 CXR HDR
FM059 HW
ITEMS
FD163 CABLE COMPL HDR
FD162 CA PAIR STATUS DETAIL
FD150 EQ DETAIL
FD154 EQ/FAC PDG ASSIGN
FD174 CXR NOTES
FD175 CXR CHAN DETAIL
FD176 CXR FAC LINES
FD173 CXR HDR
HW ITEMS
FM159
UNDER-CONSTRUCTION EQUIPMENT & FACILITIES

EXHIBIT III -2

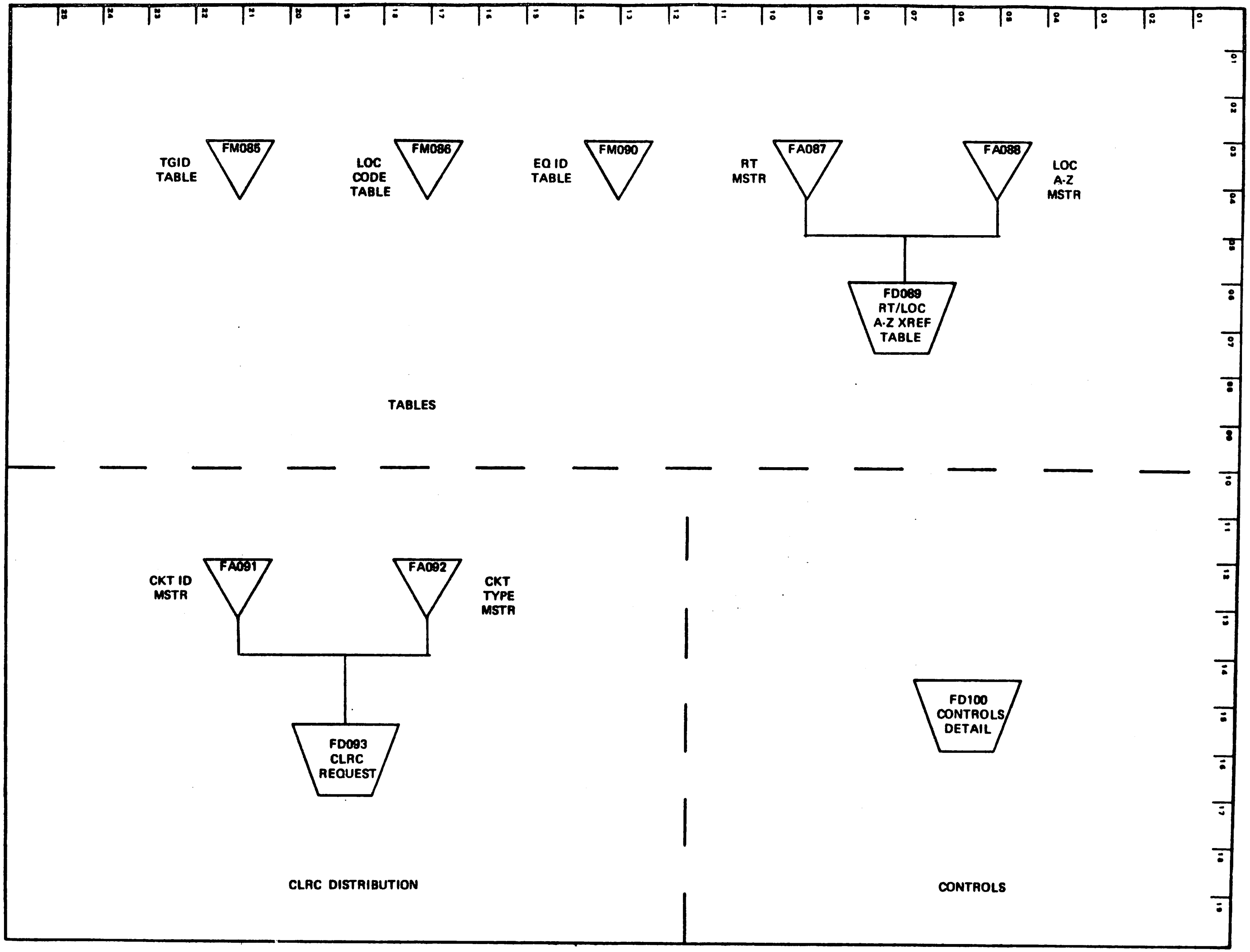

FM085
TGID TABLE
FM086
LOC CODE TABLE
FM090
EQ ID TABLE
FA087
RT MSTR
FA088
LOC A-Z MSTR
FD089 RT/LOC A-Z XREF TABLE
TABLES
FA091
CKT ID MSTR
FA092
CKT TYPE MSTR
FD093 CLRC REQUEST
CLRC DISTRIBUTION
FD100 CONTROLS DETAIL
CONTROLS

EXHIBIT III - 3

Problem: Maximum number of shared files open at one time.
Constraint: Up until the release of the 1906 MIT, only 63 shared files could be open at one time.
Resolution: This problem was brought to the attention of the HP Lab. We pointed out that a shared data base could contain 99 data sets but that only 63 of them could be open at any one time. This problem was resolved by the lab before it became a serious problem in a production environment.

Problem: Maximum number of data sets in a data base
Constraint: The maximum number of data sets in a data base is 99.
Resolution: Several attempts at data base design ended up with 75-85 data sets. We felt that this left too few data sets for future enhancements. Therefore, we included the number of data sets as a criteria for design. Other criteria used was:

1) System performance
2) Disc requirement
3) Query capability

Problem: Maximum number of entries in a chain.
Constraint: IMAGE only allows 64,000 items in a chain. We had one situation where a longer chain length would have been very beneficial for some report retrieval requirements.
Resolution: A good resolution of this problem was not found. A partial solution was to separate certain records into different data sets.

Problem: Data Base Locking Schemes
Constraint: Item level locking, as it was implemented, presented us with a difficult situation because of our requirement to change the locking table several times during the processing of a single transaction.
Resolution: This problem is still under review. There does not appear to be a foolproof way of implementing item level locking. The best we can hope for is a method which has the least probability of creating a problem and the problem must have a minimal consequence to the user. Locking at higher levels, such as data set or data base, will obviously create an intolerable performance problem.

Problem: System Back-up Methods
Constraint: DB stores and DB unloads can be very time consuming for a data base this size, particularly when the back-up medium is tape.
Resolution: Each system will be configured with an extra disc drive. This drive will be used as a serial disc for back-up and as a private volume for logging.

Problem: Lack of multiple data set retrieval capabilities from Query.
Constraint: Query can only access a single detail data set in a find command. With the relationships that exist among data elements in the various data sets, it is often necessary to access several data sets for reporting purposes.
Resolution: We are currently reviewing software packages from vendors other than HP to satisfy this requirement.

Problem: Inability to group data elements without losing their element identity.
Constraint: Sometimes data elements need to be combined to form an IMAGE item. In particular, in order to construct certain paths into data sets this capability was required.
Resolution: For IMAGE purposes, we combined the elements. If one or more of the elements could be identified as a requirement for Query purposes, then it was repeated in the data set as a single element.

Despite the problems mentioned above, the productivity rate was nearly three times the company average for a large development project. We think that this is attributable to three things:

1) The team concept and its associated techniques.
2) The fact that we had a computer dedicated for use by the development team, and
3) The reliability of HP hardware and software.

IMPLEMENTATION

Once the system was tested by the project team, it was installed in a "pilot" company for a production test. This activity began with two weeks of detailed training on the use of the system. Also, the data processing people who will be involved in providing local support were given a week of training in the DP aspects of the system. The documentation of the system consists of a system design manual (7 volumes) containing program specifications and related documentation and a users manual (3 volumes) giving specific instructions on how to use the system.

Concurrent with the training, the conversion programs were being run to convert the data base from the IBM system to the Hewlett Packard system. Only a portion of the data base was loaded for pilot purposes. This consisted of 13,000 circuits, 50,000 items of equipment, and 30,000 facilities.

During the conversion, we had verified what we had expected. Namely, that loading a large IMAGE data base is a very time consuming operation.

The cumulative processing time for the conversion was 77 hours. This time includes the periodic back-ups that were taken between jobs. Since this was a pilot area, these times were acceptable. However, when projected for converting a full company, these times would be too long. Therefore, the project team is looking at ways to improve the time. Hewlett Packard has been contracted to do a performance analysis on the conversion programs. Also, some changes in the design of the conversion method are being contemplated. We hope to obtain about a 50% improvement in the performance of the conversion programs.

At the time of this writing, the pilot activity is continuing. Very few technical problems have been identified (only one program bug) at this point. There are some administrative problems with the system that have been identified and these are being addressed as high priority enhancements for the system.

During the pilot activity, the requirements team was again called in to observe the system in operation. They then provided additional input for determining the priorities for future modules of the system.

THE FUTURE

The current plans for the system are to continue developing new modules for the system. The various telephone companies will implement the version of the system that is current when they are ready for the system, while other companies are choosing to wait until certain features that they consider as essential are available. The sequence in which the various modules will be added will be determined by an executive committee based on cost/benefit statements that are being prepared for each feature. At this point in time, ten companies have requested installation of the system.

For support of the system, we intend to have a communications capability (DS3000 on a switched network) which will enable us to access any user system for the purpose of problem determination, error corrections, and program changes. When this capability is in place, we will be able to provide very efficient service from a highly trained group. This system has many more opportunities to evolve and it should evolve in a way that is consistent with evolving technology.

CONCLUSION

This paper has touched on many aspects of system development. To pursue any of these aspects in any greater detail would require a document far beyond the scope of this paper. We feel that our methods are very traditional, but what makes this situation unique, is the way that these traditional methods were applied in today's technology.

If any reader of this paper wishes to discuss any aspect of this effort in greater detail, please feel free to contact us.

Technische Universität Berlin

PROZESSRECHNERVERBUND - ZENTRALE
Leiter: Dipl.-Ing. K. Rebensburg

Telefon:
(030)314 5154/61
Sekr.: HE 3
Einsteinufer 19
D-1000 Berlin 10
GERMANY

INTRODUCTION OF PEARL
(Process and Experiment Automation Realtime Language)
ON HEWLETT-PACKARD COMPUTER SYSTEMS

Dipl.-Ing. Jörg Grössler

Contents:

1. Programming Requirements f r Process Control
2. History of PEARL Development
3. Characteristics and Standardization of PEARL
4. Experience with PEARL
5. PEARL on HP 3000
6. PEARL on a Computer Network HP 3000 - HP 1000

Appendix A: Example of a simple PEARL Application
Appendix B: Collection of PEARL Features and Examples

Dieser Bericht veröffentlicht Ergebnisse aus einem mit Mitteln des Bundesministers für Fprschung und Technologie (Kennzeichen DV 5.505) geförderten Forschungsvorhabens des Projektes "Prozeßlenkung mit DV-Anlagen (PDV)" im Rahmen des 3. DV-Programmes der Bundesregierung.

Die Verantwortung für den Inhalt liegt ausschließlich bei den Autoren bzw. den geförderten Unternehmen.

1. Programming Requirements for Process Control

General Requirements

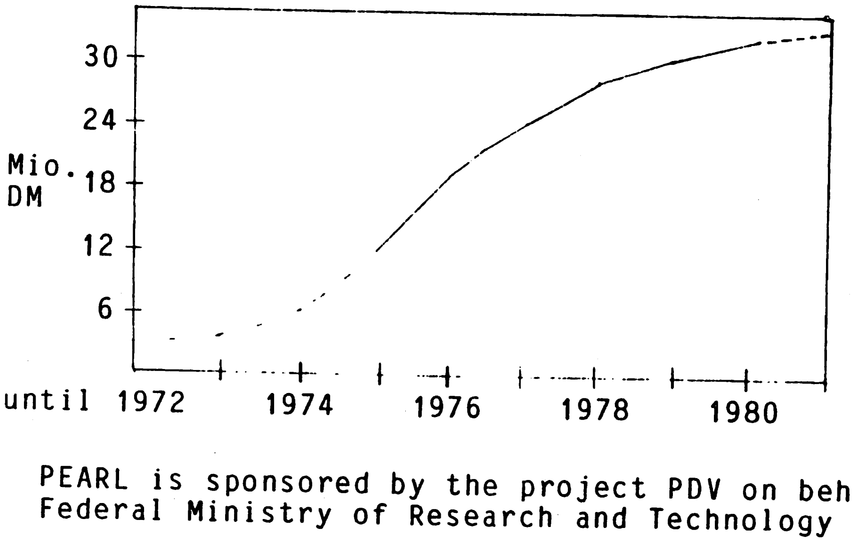
- Process control programming should not require too much knowledge of the computer internals
- The programming language should allow economical writing and testing of programs
- Source code should be easy to read
- The programming language should support portability
- Open ended design of the computer control system should be supported through a module oriented program structure
- The programming language should assist the construction of correct programs and the development of reliable control systems
- Object code should run sufficiently efficient

Special Requirements concerning Realtime Application

- Means to describe parallel activities of a program
- Possibility to describe configuration and capabilities of the system resources to be used in the program
- Special data types and operators for realtime process control

2. History of PEARL Development

1969	PEARL workshop is founded composed from the start of users, process computer manufacturers, software suppliers, research institutes, and representators of the German Society of Engeneers (VDI/VDE).
1972	Project PDV (directed by a project staff at the Nuclear Research Center, Karlsruhe, on behalf of the Federal Ministry of Research and Technology) takes PEARL under its wings.
1973	Preliminary definition of PEARL is set up.
1976	Final PEARL definition is completed.
1978	DIN Draft Standard of Basic PEARL is published.
1978	Basic PEARL is been submitted to the international working group TC97/SC5/WG1 as candidate for ISO Standard.
1979	DIN Draft Standard of Full PEARL will be published until the end of the year.



PEARL is sponsored by the project PDV on behalf of the Federal Ministry of Research and Technology

Status of PEARL Implementations:

Firm	Process Computer Family
AEG-Telefunken	80-20/40/60
BBC	DP 1000/1500
Siemens	330/340
Dietz	Mincal 621
Krupp-Atlas	EPR 1100/1300/1500
MBP	HP 3000
MBP	Siemens 404/3
announced:	
DIGITAL	DEC PDP 11-Family
Werum	Norsk Data Nord 10S
Werum	MODCOMP
TU-Berlin (Werum)	HP 21MX
announced for microcomputers:	
Dornies (GPP)	MUDAS-432
SEL (GPP)	DEC LSI 11
IITB (Werum)	Siemens 310
Uni Karlsruhe (GPP)	Zilog Z80

3. Characteristics and Standardization of PEARL

Basic Concept of PEARL

- o Algorithmic language concept mainly based on ALGOL 68 and also influenced by PASCAL
- o Syntax adapted to PL/1 where possible
- o I/O structure allows to deal with a large variety of virtual devices (called DATIONs) and communication methods
- o Abstract data types (like TIME, DURATION, and BIT-string) needed for realtime application

Principle Layout of a PEARL program

MODULE (module name);

SYSTEM;

- Description of hardware configuration (connection)
- Introduction of freely chosen names for I/O terminals and signals

PROBLEM;

Description of actions to be executed, practically independend of environment, e.g.

- Declaration/Specification of data, tasks, procedures
- Start conditions (schedules) of tasks

MODEND;

Characteristics of PEARL

Algorithmic Features

- o Block structured language with scoping rules for objects known from the ALGOL family
- o Comprehensive choice of data types and possible attributes
- o Compound type STRUCTURE (similar to the PASCAL record)
- o Communication between processes via GLOBAL objects with optional assignation protection
- o PROCEDURE declarations with parameter transfer call-by-value or call-by-reference
- o Definition of abstract data types by using TYPE and OPERATOR declaration

Realtime Features

- o Parallel processing by using TASK blocks and PRIORITY
- o Declaration of events (INTERRUPT and SIGNAL)
- o Synchronization of parallel activities by SEMAs and BOLTs

Input and Output Features

- o Data-stations (DATIONs), generalizing real or virtual peripherals or I/O channels
- o Interfaces, mapping data-stations with different properties onto each other to offer the possibility to define formatting routines (objects of type CONTROL)
- o An object of type DATION represents in general a set of one to four channels:
  - Data channel (transfers values of PEARL objects)
  - Control channel (transfers values of type CONTROL)
  - Interrupt channel (signals events of type INTERRUPT)
  - Signal channel (signals event of type SIGNAL)

Standarization of PEARL

- o DIN (German Institute for Norm Control, Berlin) is given the task to care for a PEARL language description
- o Means to describe semantics of realtime features of programming languages are based on Petri Nets
- o DIN Draft Standard for Basic PEARL written and published in June 1978
- o DIN Draft Standard for Full PEARL is scheduled for late 1979

Basic PEARL is the common and minimal subset of Full PEARL which each implementations must contain. Some of the (Full) PEARL features Basic PEARL does not have are:

- Objects of type Reference (pointers) and BOLT
- Extensibility (new data types and operators)
- LABEL variables
- Multi-dimensional STRUCTUREs
- Nested tasks (subtasks)
- Dynamic priority change
- Interfaces
- Graphic I/O

Basic PEARL has been submitted by DIN to the international working group TC97/SC5/WG1 "Programming languages for the Control of Industrial Processes (PLIP)" of ISO (International Organization for Standardization) as candidate for ISO standard. Full PEARL is to follow in due time.

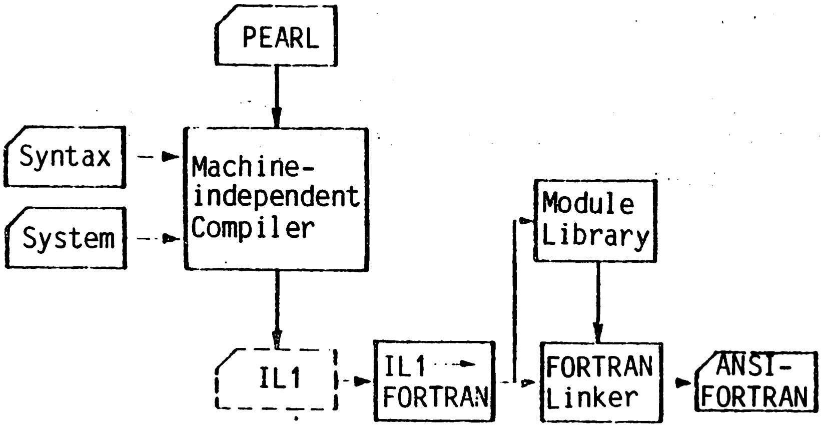
4. Experience with PEARL

- At present about 130 PEARL systems either in operation (the majority) or being installed (approx. 100 systems are truly industrial)
- Quantitative results published (up to 80 % savings)

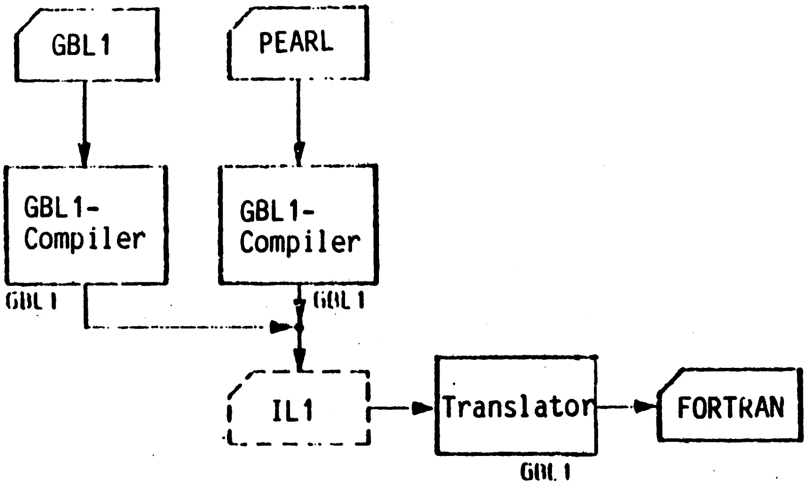
Realtime application area	Number of systems
Metallurgical plants, rolling mills	37
Power distribution	31
Power generation	4
Prime materials, chemical	16
Water supply services	10
Other public services (television, traffic, Spacelab etc.)	7
Other industrial	9
Warehouse, storage	7
PEARL R+D and education	15

5. PEARL on HP 3000

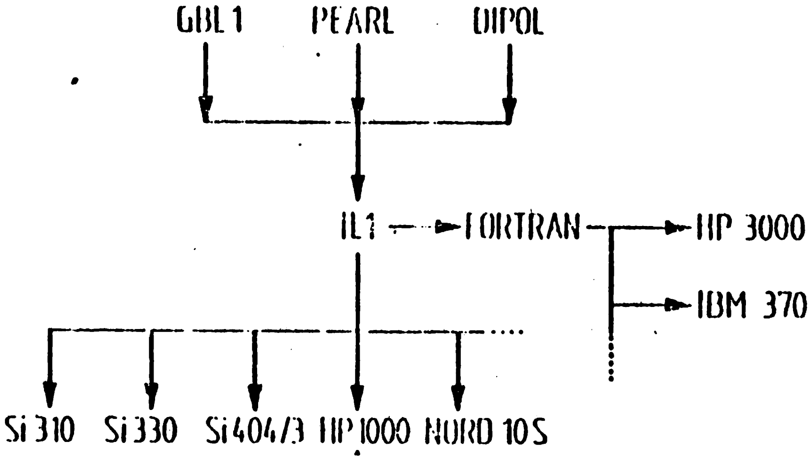
- PEARL subset on HP 3000 is Basic PEARL
  - + multi-dimensional STRUCTUREs
  - + objects of type Reference and BOLT
  - + declaration of data types and operators
- Machine-independend compiler
- Control of syntax analysis by a list to minimize expenses by changes of the language syntax
- Control of SYSTEM division interpretation by a list written in PEARL
- Runtime routines using the HP 3000 FORTRAN-library
- Universal PEARL operating system written in GBL1 (PL/1-subset)
- PEARL operating system implemented on MPE/II and MPE/III without modifications of the MPE operating system
- Three HP 3000 processes running for PEARL
  - one process containing all PEARL tasks
  - one process realizing I/O operations
  - one process for timing control
- five software interfaces betweeen MPE and the PEARL operating system
- Code generator to produce HP 3000 RBMs instead of FORTRAN announced


PEARL
Syntax
System
Machine-independent Compiler
Module Library
IL1
IL1 → FORTRAN
FORTRAN Linker
ANSI-FORTRAN

Principle of the PEARL compiler system on HP 3000

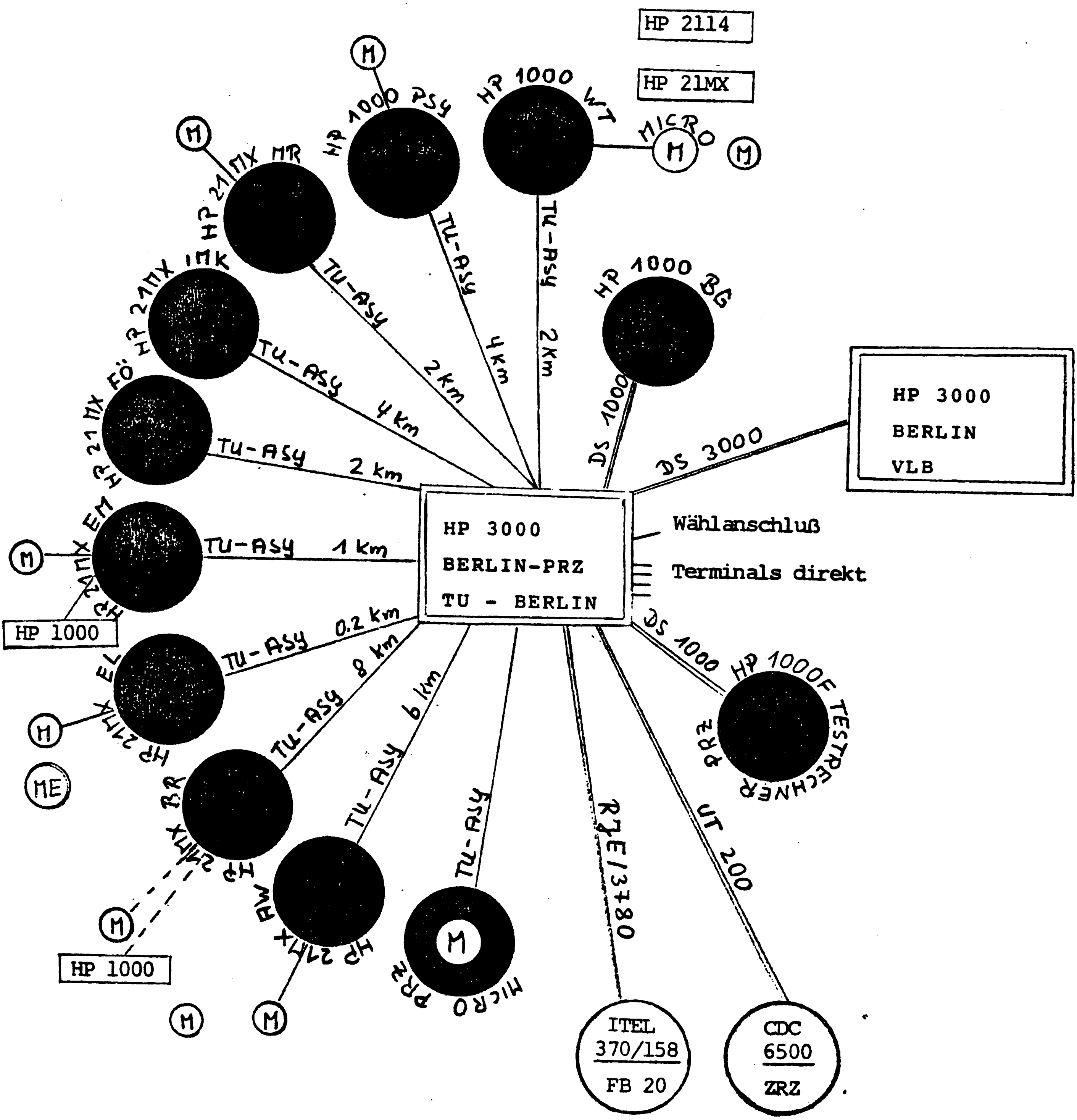

GBL1
PEARL
GBL1-
Compiler
GBL1-
Compiler
GBL1
GBL1
IL1
Translator
FORTRAN
GBL1

Method of implementing the PEARL compiler system on HP 3000


GBL1
PEARL
DIPOL
IL1
FORTRAN
HP 3000
IBM 370
Si 310
Si 330
Si 404/3
HP 1000
NORD 10S

Advantage of using machine independent compilers by generating IL1 (Intermediate language) as computer output

6. PEARL on a Computer Network HP 3000 - HP 1000


HP 2114
HP 21MX
M
HP 1000 PSY
HP 1000 WT
MICRO
M
M
M
HP 21MX MR
TU-ASY 2 km
TU-ASY 4 km
TU-ASY 2 km
HP 21MX IMK
TU-ASY 4 km
HP 1000 BG
DS 1000
HP 21MX FÖ
TU-ASY 2 km
HP 3000
BERLIN
VLB
DS 3000
M
HP 21MX EM
TU-ASY 1 km
HP 3000
BERLIN-PRZ
TU - BERLIN
Wählanschluß
Terminals direkt
HP 1000
HP 21MX EL
TU-ASY 0.2 km
M
ME
DS 1000
HP 1000F TESTRECHNER PRZ
HP 21MX BR
TU-ASY 8 km
TU-ASY 6 km
TU-ASY
RJE/3780
UT 200
M
HP 1000
HP 21MX AW
M
MICRO PRZ
M
M
ITEL
370/158
FB 20
CDC
6500
ZRZ

The Realtime-Computer-Network of the Technical University Berlin consists of 11 realtime-systems HP 21MX and HP 1000 in star-configuration with a central computer HP 3000. The satellites are used in various fields of process control like psychology, ergonomics or brewery and can have microcomputers as subcomputers. The central HP 3000 is mainly used for program development and remote file access. In order to increase the availability of the system a project has been started in 1978 to introduce PEARL into the process computer network.
The main characteristics of this PEARL implementation are:

- o Introducing PEARL into the process-computer-network is done in three phases:
  - Implementing the excisting PEARL system on the central HP 3000
  - Implementing a cross-compiler system for HP 1000 on HP 3000
  - Implementing a stand-alone compiler system on HP 1000
- o For all three phases the same compiler main part is used; a code generator is written to produce HP 21MX relocatable code
- o Sophisticated DS-features will be included to make PEARL applicable for realtime-computer-networks
- o As a prospect it is planed to implement PEARL cross software on HP 3000 for microprocessors to be configured a subsystem under each satellite computer

References:

T. Martin: Realtime Programming Language PEARL - Concepts and Characteristics
Proceedings of COMPSAC 78 page 301

Full PEARL Language Description,
PDV-Report KfK-PDV 130,
Kernforschungszentrum Karlsruhe GmbH, 1977

DIN 66253 Basic PEARL Draft Standard
Beuth Verlag GmbH, Berlin, Cologne 1978

K.Rebensburg: Real-time-computing with the Computer network of the technical university Berlin

Example of a simple PEARL application

A pipeline shall be pressure controlled and is properly instrumented for this purpose (Fig. 1).

Fig. 2 shows the function/data block structure of the planned computer control system. The next step would be to choose a hardware configuration and connect the sensors, lamp, and control devices to certain clamps of the computer I/O devices (Fig. 3).
There are two tasks: Task 1 controls the pressure by means of a simple on/off algorithm. The valve is opened and closed by outputting bit pattern "10" or "01", respectively. Task 2, in addition, takes measures to alarm and quickly release the pressure, if a certain high limit is reached. Next the PEARL program can be implemented. (In the following text capital letters are used for designating PEARL language elements.) Before this, the general program structure of PEARL must be explained. PEARL programs are composed of independently compilable units, the so called MODULEs. Connections between MODULEs are established through GLOBAL objects. In general, a MODULE consists of a part describing the hardware configuration (especially the process I/O connections), called SYSTEM division, and another one containing the formulation of the problem, called PROBLEM division:

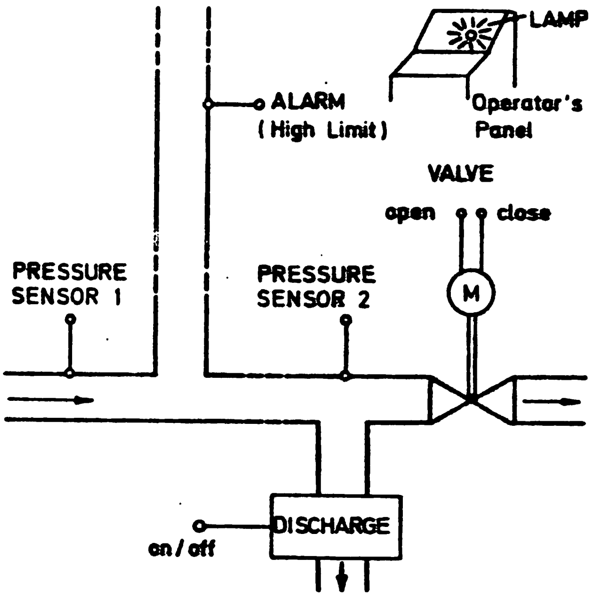

LAMP
Operator's Panel
ALARM (High Limit)
VALVE
open
close
PRESSURE SENSOR 1
PRESSURE SENSOR 2
M
DISCHARGE
on/off

Figure 1 Pressure control scheme for pipeline

```
MODULE (module name);
   SYSTEM;
       - Description of hardware
         configuration (connections)
       - Introduction of freely chosen
         names for I/O terminals (DATIONs),
         interrupts, and signals
   PROBLEM;
         Description of actions to be executed,
         practically independent of environ-
         ment, e.g.
       - Declaration/Specification of
         data, tasks, procedures
       - Start conditions (schedules)
MODEND;  of tasks
```

The purpose of this separation is the following: The PROBLEM division which contains the algorithmic and organizing part of the program becomes invariant against hardware changes. Thus, portability of the control program is achieved!
The PROBLEM division is structured by means of TASKs and PROCEDUREs. TASKs are independently and simultaneously runnable program parts. PROCEDUREs (not used in the example) are, as usual, dependent on the CALLing program part.

With this information the reader should be almost able to understand the PEARL program, given in Fig. 4, utilizing the self-documentation feature of PEARL. In the SYSTEM division the transfer direction is defined by arrow symbols. Note how the user names of the devices (i.e. DATIONs) are introduced like labels. The other hardware names are system names known by the compiler.

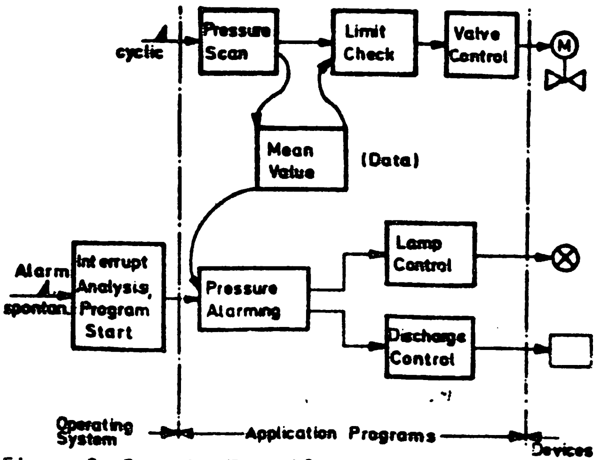

cyclic
Pressure Scan
Limit Check
Valve Control
M
Mean Value
(Data)
Alarm
spontan.
Interrupt Analysis, Program Start
Pressure Alarming
Lamp Control
Discharge Control
Operating System
Application Programs
Devices

Figure 2 Function/Data block structure of computer control system

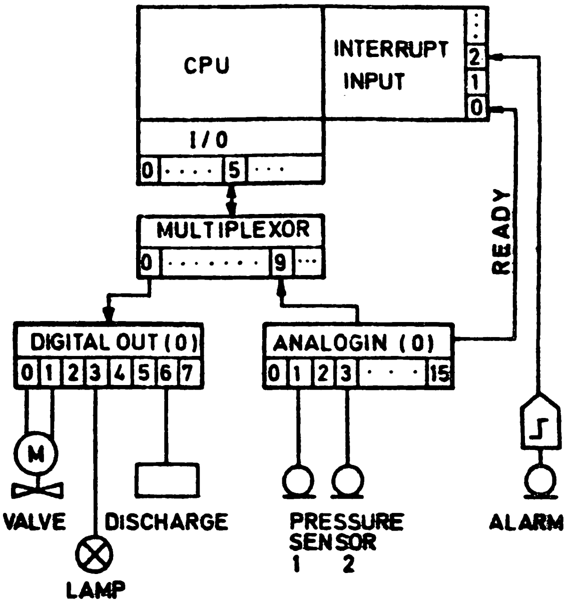


Figure 3 Computer hardware configuration with process connections

```
MODULE; /*EXAMPLE PRESSURE CONTROL*/
SYSTEM;

MULTIPLEXOR    <-->  CPU*5;
DIGITALOUT(0)  <--   MULTIPLEXOR*0;
ANALOGIN(0)     -->  MULTIPLEXOR*9;

VALVE      :      <--   DIGITALOUT(0)*0.2;
LAMP       :      <--   DIGITALOUT(0)*3.1;
DISCHARGE  :      <--   DIGITALOUT(0)*6.1;

PRESSURESENSOR1:   -->   ANALOGIN(0)*1;
PRESSURESENSOR2:   -->   ANALOGIN(0)*3;

READY:    -->   INTERRUPTINPUT*0;
ALARM:    -->   INTERRUPTINPUT*2;

PROBLEM;
SPECIFY  VALVE DATION OUT BASIC DIM() TFU,
         (LAMP, DISCHARGE) DATION OUT BASIC,
         (PRESSURESENSOR1, PRESSURESENSOR2) DATION IN BASIC DIM() TFU,
         ALARM INTERRUPT;

DECLARE (PRESS1, PRESS2, MEANPRESS) FLOAT,
        OPENVALVE BIT(2)  INITIAL('10'B),
        CLOSEVALVE BIT(2) INITIAL('01'B),
        (TURNON, TURNOFF) BIT INITIAL('1'B,'0'B);

START:           TASK GLOBAL; /*ACTIVATED FROM OUTSIDE*/
                   :
                 EVERY 1 SEC ACTIVATE PRESSURECONTROL;
                 WHEN ALARM ACTIVATE ALARMING;
                 END; /*OF TASK START*/

PRESSURECONTROL: TASK PRIORITY 5;
                 TAKE PRESS1 FROM PRESSURESENSOR1;
                 TAKE PRESS2 FROM PRESSURESENSOR2;
                 MEANPRESS = (PRESS1 + PRESS2)/2;
                 IF MEANPRESS >= 30
                    THEN SEND OPENVALVE TO VALVE;
                    ELSE IF MEANPRESS <20 THEN SEND CLOSEVALVE TO VALVE;
                       FIN;
                 FIN;
                 END; /*OF TASK PRESSURECONTROL*/

ALARMING:        TASK PRIORITY 3;
                 SEND TURNON TO LAMP;
                 SEND TURNON TO DISCHARGE;
                 WHILE MEANPRESS >= 20
                    REPEAT;
                    AFTER 2 MIN RESUME;
                 END; /*OF LOOP*/
                 SEND TURNOFF TO LAMP;
                 SEND TURNOFF TO DISCHARGE;
                 END; /*OF TASK ALARMING*/
MODEND;
```

Figure 4 PEARL source program

Collection of PEARL Features and Examples

A1. PEARL OBJECTS (DATA TYPES)

(1) PROBLEM DATA

Integer	FIXED (Precision)
Floating Point	FLOAT (Precision)
Bit String	BIT (Length)
Character String	CHAR (Length)
Time Interval	DURATION
Point in Time	CLOCK

(2) DATA FOR PROCESS CONTROL

Task	TASK
Interrupt } Events	INTERRUPT
Signal	SIGNAL
Semaphore	SEMA
Bolt	BOLT

(3) DATA FOR I/O CONTROL

File Device } Data Station	DATION
Standard Formats	F, E, A, B, T, D, LIST, X, SKIP, PAGE, etc.
C-Channel Value	CONTROL
Remote Format	FORMAT
User-Defined Format	INTFAC

(4) DATA REFERENCE

Pointer	REF

(5) COMPOUND OBJECTS

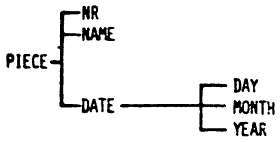
Array	Ident. (bound-pair-list)
Structure (Data Hierarchy)	STRUCTURE

(6) USER-DEFINED DATA TYPES

TYPE

A2. EXAMPLES OF PEARL OBJECTS

```
DECLARE I FIXED,
        (F,G) FLOAT,
        B BIT (16) INIT ('OOFF'B4),
        C CHAR (4) INIT ('ABCD'),
        A (1:10,-5:5) FLOAT,
        RF REF FLOAT,
        TAG LABEL RANGE(M1, M2),
        PIECE STRUCT
             [NR FIXED (10),
              NAME CHAR (8),
              DATE STRUC
                  [(DAY, MONTH, YEAR) FIXED (2)
                  ]
             ];
```



Component access: PIECE.NAME
PIECE.DATE.YEAR

(String Selection similiar)

A3. TRANSFER-OF-CONTROL STATEMENTS

Block

```
BEGIN; Local Declarations
       Statements
END;
```

Conditional-Statement (Alternative)

```
IF Bit-one-Expression
   THEN Statements 1
   ELSE Statements 2
FIN;
```

Case-Statement (Selection)

```
CASE Expression
   ALT Statements 1
   .
   .
   .
   ALT Statements N
   OUT Statements (Error Exit)
FIN;
```

Repeat-Statement (Iteration)

```
FOR Identifier (Loop Index)
FROM     Expression (Initial Value)
BY       Expression (Step Width)
TO       Expression (End Value)
WHILE    Bit-one-Expression
REPEAT   Local Declarations
         Statements
END;
```

A4. TASKING AND SCHEDULING

TASK DECLARATION

```
T:  TASK PRIO I;
    Local Declarations
    Statements
    END;
```

TASK EXECUTION

at once

```
ACTIVATE T;
ACTIVATE T PRIO (3*I);
```

at point in time

```
AT 8:30:0  ACTIVATE T;
DCL TIME CLOCK,
    PERIOD DUR;
.
.
.
TIME    = 9:0:0;
PERIOD  = 2 HRS;
.
.
.
```

periodically

```
AT TIME ALL PERIOD DURING 12 HRS
ACTIVATE T;
```

at interrupt

```
DCL FULL INTERRUPT;
.
.
.
.
WHEN FULL ACTIVATE T;
```

after 10 sec

```
WHEN FULL AFTER 10 SEC ACTIVATE T;
```

periodically for 5 min

```
WHEN FULL ALL 2 SEC DURING 5 MIN
ACTIVATE T;
```

Delay for Time Span	AFTER 5 SEC RESUME;
Delay till Interrupt	WHEN MESSAGE RESUME;
Temporary Stop	SUSPEND; or SUSPENDT T;
Continue	CONTINUE T PRIO 3;
Kill Activity	TERMINATE; or TERMINATE T;
Remove all Schedules Pending	PREVENT T;

Note: Appendix A and B are taken form "Realtime Programming Language PEARL-Concepts and Characteristics" by T. Martin (see references).

" The next IT " - Ray Johnson - HP CUPERTINO

1918 MPE RELEASE

MAJOR ENHANCEMENTS:

- * MPE USER LOGGING
- * CONSOLE ENHANCEMENTS/DISTRIBUTED CONSOLE
- * SPOOLER ENHANCEMENTS
- * STORE/RESTORE TAPE LABELS
- * VIRTUAL MEMORY INCREASE
- * MPE RESEGMENTATION

MPE USER LOGGING

- ALLOWS USER TO INTERFACE LOGGING TO ANY SUBSYSTEM USING NEW LOGGING INTRINSICS (LOGGING AND RECOVERY ROUTINES HAVE BEEN ADDED TO IMAGE/3000)

- LOG RECORDS MAY BE WRITTEN TO A DISC FILE OR TO A TAPE

MPE USER LOGGING

USER COMMANDS

NOTE: LG CAPABILITY REQUIRED FOR ALL BUT :SHOWLOGSTATUS

- :ALTLOG ALTERS THE ATTRIBUTES OF AN EXISTING LOGGING IDENTIFIER

```
                            ┌,DISC┐
:ALTLOG LOGID [;LOG=LOGFILE {     }] [;PASS=PASSWORD]
                            └,TAPE┘
```

- :GETLOG ESTABLISHES A LOGGING IDENTIFIER ON THE SYSTEM

```
                           ┌,DISC┐
:GETLOG LOGID ;LOG=LOGFILE {     }  [;PASS=PASSWORD]
                           └,TAPE┘
```

- :LISTLOG LISTS LOGGING IDENTIFIERS

```
:LISTLOG [LOGID [;PASS] ]
```

- :RELLOG REMOVES A LOGGING IDENTIFIER FROM THE SYSTEM

```
:RELLOG LOGID
```

- :SHOWLOGSTATUS DISPLAYS STATUS INFORMATION ABOUT CURRENTLY RUNNING LOG PROCESSES

```
:SHOWLOGSTATUS [LOGID]
```

MPE USER LOGGING

OPERATOR COMMANDS

* :LOG STARTS, RESTARTS, OR STOPS USER LOGGING

```
               {START}
:LOG LOGID,    {RESTART}
               {STOP }
```

MPE USER LOGGING

USING THE FACILITY

1. MAKE CERTAIN USER LOGGING CAPABILITY HAS BEEN ADDED TO YOUR CAPABILITY SET (LG)
2. CHECK TO SEE THAT THE REQUIRED NUMBER OF LOGGING PROCESES AND USERS PER PROCESS HAVE BEEN ALLOWED (NEW QUESTIONS IN SYSDUMP DIALOG)

ESTABLISHING A LOGGING FILE:

IF LOGGING TO DISC, YOU MUST BUILD A LOGGING FILE:

```
:BUILD NEWDATA;CODE=LOG
```

1090 -- L[illegible]

ESTABLISHING A LOGGING IDENTIFIER:

USED TO IDENTIFY THE LOGGING PROCESS THAT WRITES THE TRANSACTIONS RECORDS TO THE LOGGING FILE

TO ESTABLISH THE LOGGING IDENTIFIER FINANCE AND ASSOCIATE IT WITH A DISC LOGGING FILE NAMED NEWDATA, ENTER:

```
:GETLOG FINANCE;LOG=NEWDATA,DISC
```

MPE USER LOGGING

USING THE FACILITY

ACTIVATING USER LOGGING:

USER LOGGING IS TURNED ON WITH THE NEW CONSOLE OPERATOR COMMAND :LOG

```
:LOG FINANCE,START
```

START: INITIATES LOGGING PROCESS.
RESTART: RESTARTS LOGGING PROCESS
STOP: TERMINATES LOGGING PROCESS

MODIFYING A LOGGING IDENTIFIER:

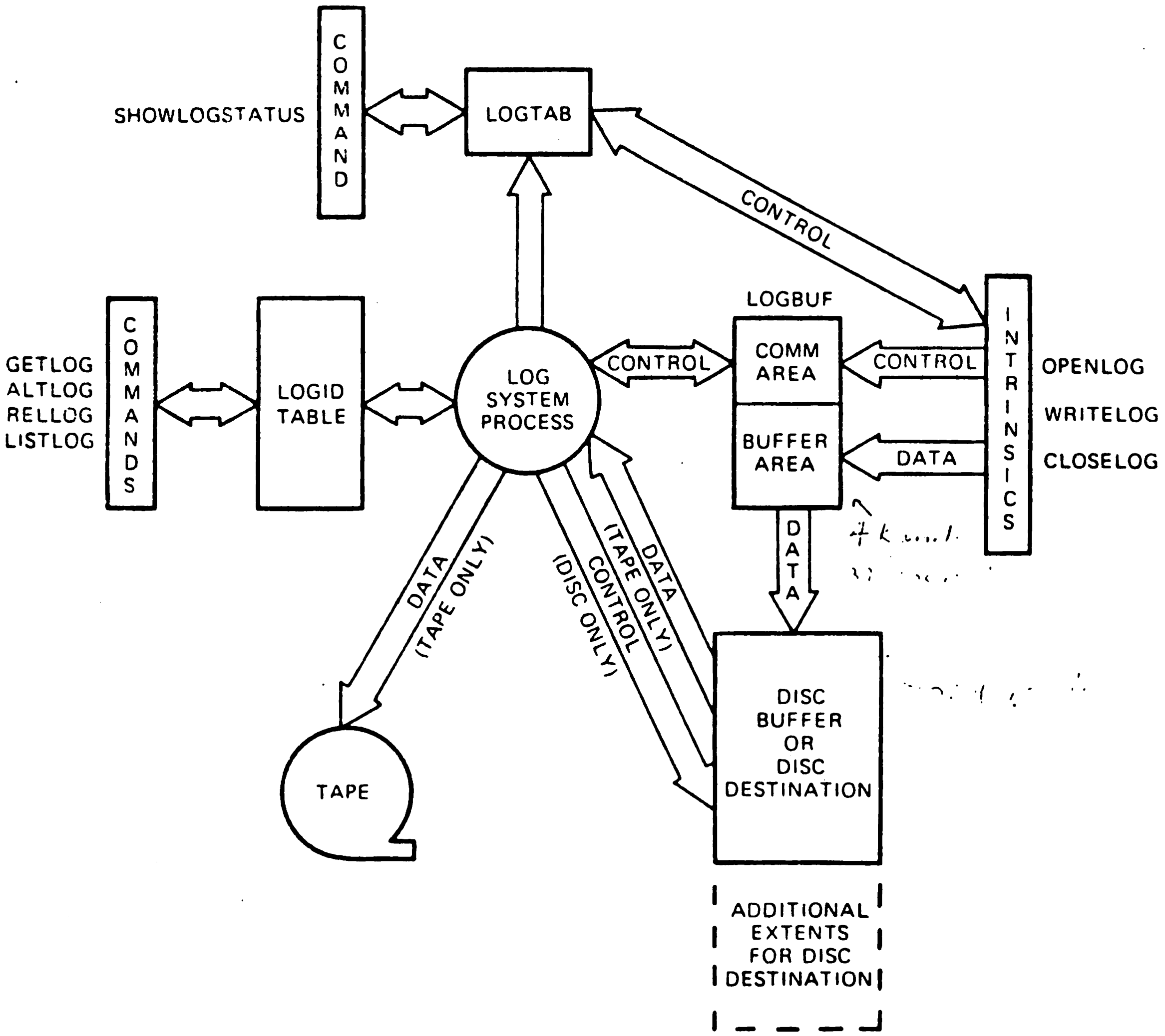
THE :ALTLOG COMMAND IS USED TO ALTER THE LOGGING FILE OR PASSWORD ASSIGNED TO AN EXISTING LOGGING IDENTIFIER

```
:ALTLOG FINANCE;LOG=FILEM,TAPE
```

WRITING TO THE LOG FILE:

LOGGING IS DONE PROGRAMMATICALLY BY MEANS OF THREE INTRINSICS:

OPENLOG, WRITELOG, AND CLOSELOG


SHOWLOGSTATUS
COMMAND
LOGTAB
CONTROL
LOGBUF
GETLOG
ALTLOG
RELLOG
LISTLOG
COMMANDS
LOGID TABLE
LOG SYSTEM PROCESS
CONTROL
COMM AREA
CONTROL
BUFFER AREA
DATA
INTRINSICS
OPENLOG
WRITELOG
CLOSELOG
DATA (TAPE ONLY)
DATA (TAPE ONLY)
CONTROL (DISC ONLY)
DATA
TAPE
DISC BUFFER OR DISC DESTINATION
ADDITIONAL EXTENTS FOR DISC DESTINATION

User Logging Facility

USER LOGGING RECORD FORMATS

D. LOGGING RECORD FORMAT

RECORD SIZE = 128 WORDS
USER AREA = 119 WORDS

2. LOG RECORD AT OPENLOG

```
0      2     3     4     6     7      11    12        24            127
------------------------------------------------------------------------
|  1  |     |     |     |     |      |     |         |    |          |
| rec#|cksum|code |time |date | logid|log# |  creator|pcb |          |
|-----|-----|-----|-----|-----|------|-----|---------|----|----------|
```

USER OR SUBSYSTEM/CONTINUATION LOG RECORD

```
0      2     3     4     6     7     8                                127
------------------------------------------------------------------------
|  2  |     |     |     |     |     |    |                            |
| rec#|cksum|code |time |DATE |log# |len |         user area          |
|-----|-----|-----|-----|-----|-----|----|----------------------------|
```

LOG RECORD AT CLOSELOG

```
0      2     3     4     6     7      11    12        24            127
------------------------------------------------------------------------
|     |     |     |     |     |      |     |         |    |          |
| rec#|cksum|code |time |date | logid|log# |  creator|pcb |          |
|-----|-----|-----|-----|-----|------|-----|---------|----|----------|
```

CRASH MARK

```
0      2     3     4     6                                          127
------------------------------------------------------------------------
|     |     |     |     |     |                                       |
| rec#|cksum|code |time |date |                                       |
|-----|-----|-----|-----|-----|---------------------------------------|
```

1. HEADER RECORD START/RESTART

```
0      2     3     4     6     7         11                         127
------------------------------------------------------------------------
|  0  |     |     |     |     |         |                             |
| rec#|cksum|code |time |date | LOGID   |                             |
|-----|-----|-----|-----|-----|---------|-----------------------------|
```

USER LOGGING RECORD FORMATS

TRAILER RECORD

```
0     2     3     4    6                                          127
 -----------------------------------------------------------------------
|     |     |     |    |    |                                        |
| rec#|cksum|code |time|date|                                        |
|-----|-----|-----|----|----|----------------------------------------|
```

NULL RECORD

```
0     2     3     4    6                                          127
 -----------------------------------------------------------------------
|     |     |     |    |    |                                        |
| rec#|cksum|code |time|date|                                        |
|-----|-----|-----|----|----|----------------------------------------|
```

CODE DEFINITION

CODE=1	OPEN LOG
2	USER/SUBSYSTEM RECORD
3	CLOSE LOG
4	HEADER
5	TRAILER
6	RESTART
7	CONTINUATION OF USER OR SUBSYSTEM RECORD
9	CRASH MARKER
SPACE	NULL RECORD (8442 or %20040)

NOTE:

1. The checksum algorithm uses the exclusive or function against a base of negative one. All records are included except the checksum.

2. Null record is used for filler.

3. The code word of the logging record can contain a subsystem code defined by the user in the first half of the word (0:8). User logging allows privileged users to pass this code in the index parameter of the Open Log intrinsic.

USER LOGGING

POWER FAIL RECOVERY

IF POWER FAIL OCCURS WHILE LOGGING TO MAG TAPE, THE FOLLOWING RECOVERY PROCEDURE MUST BE USED TO RECOVER AND CONTINUE LOGGING.

1. AFTER POWER IS BACK ON AND THE SYSTEM IS RUNNING THE CONSOLE WILL RECEIVE A NOT READY MESSAGE FOR THE TAPE

2. PRESS THE FOLLOWING BUTTONS ON THE TAPE DRIVE IN THE GIVEN ORDER:

   A. LOAD
   B. RESET
   C. ONLINE

3. AFTER THE ONLINE BUTTON IS PRESSED THE TAPE WILL MOVE BACK TO THE BEGINNING OF DATA AND MOVE FORWARD TO RECOVER THE LOG RECORDS. THEN THE SYSTEM WILL CONTINUE LOGGING.

USER LOGGING

SYSTEM FAILURE RECOVERY

- MUST WARMSTART SYSTEM

- ANY DATA IN LOGGING BUFFER AREA AT TIME OF FAILURE WILL BE LOST

- FOR DISC LOG FILES, WARMSTART RECOVERY PROCEDURE READS DISC FILE UNTIL LAST BLOCK OF DATA IS READ;

  PROCEDURE WRITES OUT CRASH RECORD AND CLOSES FILE

- FOR TAPE LOG FILES, PROCEDURE OPENS DISC BUFFER AND TAPE FILE AND COMPARES LAST RECORDS;

  PROCEDURE WRITES OUT DATA IN DISC BUFFER THAT IS NOT ON TAPE TO THE TAPE;

  PROCEDURE WRITES OUT CRASH RECORD AND CLOSES FILE

- TO RESTART LOGGING, :LOG logid, RESTART MUST BE ISSUED

MPE USER LOGGING

IMAGE: TRANSACTION LOGGING AND RECOVERY SYSTEM

IMAGE/3000 PROVIDES A LOGGING AND RECOVERY SYSTEM THAT CAN BE EXECUTED TO BRING DATA BASES BACK TO A SEMBLANCE OF THEIR STATE AT THE TIME OF A SYSTEM FAILURE.

* THE LOGGING SYSTEM PROVIDES A MECHANISM TO LOG DATA BASE TRANSACTIONS TO A LOGFILE ON TAPE OR DISC

* THE RECOVERY SYSTEM READS THIS LOGFILE TO RE-EXECUTE TRANSACTIONS AGAINST A DATA BASE BACKUP COPY IN THE EVENT OF A FAILURE

* THE LOGFILE MAY ALSO BE USED TO AUDIT MODIFICATIONS TO ITEMS IN THE DATA BASE

CONSOLE ENHANCEMENTS/DISTRIBUTED CONSOLE

- OPERATOR.SYS AUTOMATICALLY LOGGED ONTO SYSTEM
- CONSOLE COMMANDS ARE HANDLED BY THE COMMAND INTERPRETER
- THE CONSOLE CAN BE MOVED TO ANY TERMINAL
- THE CONSOLE OPERATOR (MASTER OPERATOR) CAN ASSIGN INDIVIDUAL USERS THE ABILITY TO EXECUTE SPECIFIC CONSOLE COMMANDS FROM THEIR TERMINALS
- USERS CAN BE ASSOCIATED WITH SPECIFIC DEVICES, ALLOWING THEM TO USE COMMANDS THAT CONTROL THE DEVICES
- USERS CAN BE GRANTED THE ABILITY TO USE JOB CONTROL COMMANDS ON THEIR OWN JOBS

OPERATOR.SYS

- * USER OPERATOR.SYS AUTOMATICALLY LOGGED ON AT SYSTEM STARTUP BECAUSE MANY CONSOLE COMMANDS REQUIRE SESSION
- * ALLOWS LOG ON UDC'S - STREAMS 10, VMOUNT ON,AUTO, ETC.
- * AFTER INSTALLATION OF 1918 IT, USER "OPERATOR" MUST BE CREATED [illegible]
- * OPERATOR.SYS SHOULD BE GIVEN THE FOLLOWING CAPABILITIES:

  IA, BA, SF, ND, UV, OP,LG

OPERATOR.SYS

```
HP32002B.01.00
WHICH OPTION <WARMSTART/COOLSTART>? COOL
ANY CHANGES?

DATE (M/D/Y)?7/6/79
TIME (H:M)?9:57
FRI, JUL  6, 1979,  9:57 AM? (Y/N)Y
LOG FILE NUMBER 1868 ON
*WELCOME*
:HELLO OPERATOR.SYS:HIPRI
9:57/15/SP#6/SPOOLED OUT
9:57/#S1/16/LOGON FOR: OPERATOR.SYS,PUB ON LDEV #20
HP3000 / MPE III B.01.00.  FRI, JUL  6. 1979,  9:57 AM
*** MORRIS ***
STREAMS 10
HEADOFF 6
VMOUNT ON,AUTO
ALLOCATE EDITOR
ALLOCATE SPL
```

DISTRIBUTED CONSOLE COMMANDS

:CONSOLE

:ALLOW
:DISALLOW

:ASSOCIATE
:DISASSOCIATE

:JOBSECURITY

DISTRIBUTED CONSOLE COMMANDS

CONSOLE COMMAND		EFFECT
:CONSOLE	→	CONSOLE TRANSFERRED TO SPECIFIC TERMINAL
:ALLOW	→	SPECIFIC CONSOLE COMMANDS GIVEN TO SPECIFIC USERS
:JOBSECURITY	→	ALL USERS GIVEN JOB CONTROL COMMANDS FOR THEIR OWN JOBS

USER COMMAND		EFFECT
:ASSOCIATE	→	CONSOLE COMMANDS FOR SPECIFIC DEVICES GIVEN TO SPECIFIC USERS

DISTRIBUTED CONSOLE COMMANDS

:CONSOLE

* CHANGES THE SYSTEM CONSOLE FROM ITS CURRENT DEVICE TO ANOTHER JOB-ACCEPTING (NON-DS) TERMINAL

FORMAT:

```
:CONSOLE  LDN
```

COMMAND ENVIRONMENT:

* USE THE :CONSOLE COMMAND WITH EXTREME CAUTION *

EXAMPLE:

TO CHANGE THE SYSTEM CONSOLE TO LDEV 23, ENTER:

```
:CONSOLE 23
```

NOTE: COLD LOADING THE SYSTEM CAUSES THE SYSTEM TO COME UP WITH THE CONSOLE AS ORIGINALLY CONFIGURED

DISTRIBUTED CONSOLE COMMANDS

:ALLOW

- THROUGH THE :ALLOW COMMAND, THE CONSOLE OPERATOR HAS THE ABILITY TO DISTRIBUTE CONSOLE COMMAND CAPABILITIES AMONG SYSTEM USERS BY DIRECTLY ENTERING:

```
        { @.@       }
:ALLOW  { USER.@    }; COMMANDS= COMMAND 1[,COMMAND 2,..COMMANDN]
        { @.ACCT    }
        { USER.ACCT }
```

EXAMPLE:

```
:ALLOW USER.STUDENT;COMMANDS = REPLY,ABORTIO
```

THIS ALLOWS USER.STUDENT TO EXECUTE :REPLY AND :ABORTIO

- THE :ALLOW COMMAND CAN ALSO BE USED IN AN INDIRECT MODE. IN THE INDIRECT MODE, AN EDITOR FILE IS CREATED CONTAINING:

```
{ @.@       }
{ USER.@    }; COMMANDS= COMMAND 1[,COMMAND 2,..COMMANDN]
{ @.ACCT    }
{ USER.ACCT }
```

THE :ALLOW COMMAND IS THEN EXECUTED BY ENTERING:

```
:ALLOW FILE= FILENAME [;SHOW]
```

- THE :ALLOW COMMAND MAY BE USED IN SUBSYSTEM MODE AS FOLLOWS:

```
:ALLOW
>USER.STUDENT;COMMANDS = REPLY,ABORTIO
>EXIT
```

:NOTE: THE USER BEING ALLOWED MUST BE LOGGED ON, AND IS DISALLOWED AT LOGOFF.

****
* CAUTION *
****

THE :ALLOW COMMAND IS EXTREMELY POWERFUL. THE MASTER OPERATOR MUST BE CAREFUL IN ASSIGNING COMMANDS TO USERS.

FOR EXAMPLE, IF THE :CONSOLE COMMAND OR THE :ALLOW COMMAND IS ALLOWED TO A USER, THAT USER CAN "STEAL" THE CONSOLE.

NOTE: THE CAPABILITY TO "STEAL" THE CONSOLE HAS ALWAYS BEEN AVAILABLE--- PRIV. MODE DEBUG

DISTRIBUTED CONSOLE COMMANDS

:DISALLOW

- THE :DISALLOW COMMAND PROHIBITS CERTAIN USERS FROM USING THE NAMED COMMANDS

- LIKE THE :ALLOW COMMAND, THE :DISALLOW COMMAND MAY BE USED DIRECTLY, INDIRECTLY, OR IN SUB-SYSTEM MODE.

- A USER WHO HAS BEEN ALLOWED A COMMAND IS AUTOMATICALLY DISALLOWED AT LOGOFF

EXCEPTION

IF @.@ WAS SPECIFIED AS USER SUBSET IN :ALLOW COMMAND, THE LISTED COMMANDS WILL REMAIN AS SIMPLE USER COMMANDS UNTIL :DISALLOW @.@;------ IS ISSUED

DISTRIBUTED CONSOLE COMMANDS

:ASSOCIATE

- THE :ASSOCIATE COMMAND IS A USER COMMAND WHICH ALLOWS A USER TO BECOME THE OPERATOR FOR A SPECIFIC DEVICE CLASS

SYNTAX:

```
:ASSOCIATE DEVICECLASSNAME
```

FOR EXAMPLE, A SPECIFIC LINE PRINTER CAN NOW BE ASSOCIATED WITH A USER IN THE GROUP THAT ACCESSES IT. THE ASSOCIATED USER WILL GET THE OPERATOR MESSAGES FOR THE PRINTER AND WILL BE ABLE TO USE THE OPERATOR COMMANDS FOR THE DEVICE.

- THE SYSTEM MANAGER MUST FIRST CREATE A DEVICE-USER ASSOCIATION TABLE BY RUNNING THE UTILITY "ASOCTABL.PUB.SYS"

THE UTILITY WILL PROMPT THE SYSTEM MANAGER TO ENTER THE ASSOCIATION STATEMENTS:

```
DEVICECLASSNAME=USER.ACCT [,..., USER.ACCT]
```

UPON EXITING THE UTILITY, THE RESULTING TABLE WILL BE SAVED AS ASOCIATE.PUB.SYS

NOTE: AN EDITOR FILE MAY BE USED BY THE ASOCTABL UTILITY BY MEANS OF A FILE EQUATION

EXAMPLE:

```
:FILE INPUT= FILENAME
:RUN ASOCTABL
```

NOTE: UNTIL 1936 A FILE EQUATION -:FILE INPUT=$STDIN- WILL BE NEEDED BEFORE ENTERING DATA INTO ASOCIATE.PUB.SYS FROM A TERMINAL.

:ASSOCIATE (CONT)

- * ANY USERS LISTED IN ASOCIATE.PUB.SYS MAY NOW USE THE :ASSOCIATE COMMAND FOR THE APPROPRIATE DEVICE CLASS. ALL MESSAGES FOR THAT DEVICE CLASS WILL COME TO THE FIRST USER ISSUING THE :ASSOCIATE COMMAND, AND THE USER WILL BE THE ONLY USER ON THE SYSTEM ABLE TO ISSUE CONSOLE COMMANDS FOR THE DEVICE CLASS. (WITH THE EXCEPTION OF THE MASTER OPERATOR, or ALLowed [illegible] WHO WILL BE ASKED IF HE WANTS TO OVERRIDE THE ASSOCIATED USER)

NOTE: ONLY ONE USER MAY BE ASSOCIATED WITH A DEVICE AT A TIME. AT USER LOG-OFF, DEVICE CONTROL RETURNS TO THE CONSOLE.

DISTRIBUTED CONSOLE COMMANDS

:ASSOCIATE (CONT)

* THE OPERATOR COMMANDS WHICH ARE MADE AVAILABLE TO USERS THROUGH THE :ASSOCIATE COMMAND ARE:

ABORTIO
ACCEPT
ALTSPOOLFILE
DELETESPOOLFILE
DOWNLOAD
GIVE
TAKE
SUSPENDSPOOL

HEADOFF
REFUSE
REPLY
RESUMESPOOL
STARTSPOOL
STOPSPOOL
HEADON

:ASSOCIATE

EXAMPLE

```
:HELLO BOB.STAMPS

:FILE INPUT=$STDIN
:RUN ASOCTABL.PUB.SYS

>LP=BOB.STAMPS
>EXIT

+-F-I-L-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y+
!  FILE NAME IS ASOCIATE.PUB.SYS                   !
!  FOPTIONS: NEW,B,*FORMAL*,F,N,FEQ,T              !
!  AOPTIONS:   IN/OUT,SREC,NOBLOCK,EXC,BUFFER      !
!  DEVICE TYPE: 0       DEVICE SUBTYPE: 4          !
!  LDEV: 2         DRT: 4         UNIT: 1          !
!  RECORD SIZE: 13      BLOCK SIZE: 117   (WORDS)  !
!  EXTENT SIZE: 15      MAX EXTENTS: 8             !
!  RECPTR: 7            RECLIMIT: 1023             !
!  LOGCOUNT: 258           PHYSCOUNT: 31           !
!  EOF AT: 256          LABEL ADDR: %00200000215   !
!  FILE CODE: 0       ID IS BOB       ULABELS: 0   !
!  PHYSICAL STATUS: 1000000000000001               !
!  ERROR NUMBER: 93     RESIDUE: 117      (WORDS)  !
!  BLOCK NUMBER: 31            NUMREC: 9           !
+--------------------------------------------------+
ASOCIATE.PUB.SYS- SECURITY VIOLATION  (FSERR 93
```

ASSOCIATE - EXAMPLE

```
:HELLO MANAGER.SYS

:FILE INPUT=$STDIN
:RUN ASOCTABL

>LP=BOB.STAMPS
>EXIT

END OF PROGRAM
:LISTF ASOCIATE,2
ACCOUNT=  SYS          GROUP=  PUB

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF       LIMIT R/B  SECTORS #X MX

ASOCIATE           13W  FB         256        1023   9       30  2  8

:HELLO BOB.STAMPS

:ASSOCIATE LP
:ASSOCIATE TERM

ASSOCIATING THIS DEVICE REQUIRES SYSTEM MANAGER'S PERMISSION. (CIERR 3059
```

:ASSOCIATE - EXAMPLE

```
                                                              ↓
:SHOWDEV
LDEV  AVAIL      OWNERSHIP        VOLID                  ASSOCIATION

  1   DISC       10 FILES
  2   DISC       2 FILES
  3   DISC       0 FILES
  4   DISC       0 FILES
  6   SPOOLED    SPOOLER OUT                             #S13-LP
  7   AVAIL
  8   AVAIL
  9   AVAIL
 10 A SPOOLED    SPOOLER IN
 15   AVAIL
 16   AVAIL
 20 A UNAVAIL    #S2: 2 FILES
 21 A AVAIL
 22 A AVAIL
 23 A AVAIL
 24 A AVAIL
 25 A AVAIL
 26 A UNAVAIL    SYS
 27 A AVAIL
 28 A AVAIL
 29 A AVAIL
 30 A AVAIL
 31 A AVAIL
 32 A AVAIL
 33 A AVAIL
 34 A AVAIL
 35 A AVAIL
 36 A AVAIL
 37 A AVAIL
 38 A AVAIL
 39 A UNAVAIL    #S14: 2 FILES
 40 A AVAIL
 41 A AVAIL
 42 A UNAVAIL    #S6: 2 FILES
 43 A UNAVAIL    #S7: 2 FILES
 44 A UNAVAIL    #S13: 2 FILES
 45 A AVAIL
 46 A AVAIL
 47 A AVAIL
```

DISTRIBUTED CONSOLE COMMANDS

:DISASSOCIATE

- REMOVES A DEVICE CLASS FROM THE CONTROL OF A USER

SYNTAX

```
:DISASSOCIATE DEVICECLASSNAME
```

- WHEN A USER DISASSOCIATES A DEVICE, THE OPERATOR IS NOTIFIED

NOTE: THE MASTER OPERATOR MAY REMOVE CONTROL FROM THE USER BY ISSUING THE DISASSOCIATE COMMAND HIMSELF.

AT LOG OFF, THE USER IS AUTOMATICALLY DISASSOCIATED.

DISTRIBUTED CONSOLE COMMANDS

:JOBSECURITY

- CONTROLS CAPABILITY OF USERS TO USE JOB-RELATED CONSOLE COMMANDS ON THEIR OWN JOBS

- SYNTAX:

```
:JOBSECURITY {HIGH}
             {LOW }
```

  - WHEN SET LOW, ALL USERS MAY USE THE COMMANDS ABORTJOB, ALTJOB, BREAKJOB, AND RESUMEJOB ON THEIR OWN JOBS

    ALSO, ACCOUNT MANAGERS WILL BE ABLE TO USE THESE COMMANDS ON ANY JOB IN THEIR ACCOUNT

  - WHEN SET HIGH, THESE COMMANDS MAY ONLY BE EXECUTED FROM THE CONSOLE (UNLESS INDIVIDUALLY ALLOWED)

  - DEFAULT AT SYSTEM GENERATION TIME IS HIGH, UNLESS A WARMSTART WAS USED, IN WHICH CASE JOBSECURITY WILL REMAIN AS IT WAS PRIOR TO THE WARMSTART.

COMMAND INTERPRETER CONSOLE COMMANDS

* THE FOLLOWING COMMANDS ARE STILL "CNTL A" COMMANDS:

=LOGOFF
=LOGON
=SHUTDOWN
=DSLINE
=MPLINE
=MRJE

* ALL OTHER CONSOLE COMMANDS ARE NOW HANDLED BY THE COMMAND INTERPRETER. ACCORDINGLY, BETTER ERROR MESSAGES AND THE HELP SUBSYSTEM WILL BE AVAILABLE FOR CONSOLE COMMANDS.

REDUCED CODE IN PROGEN — BANK 0

* =MOUNT AND =DISMOUNT HAVE BEEN REPLACED WITH :LMOUNT AND :LDISMOUNT BECAUSE OF EXISTING :MOUNT AND :DISMOUNT USER COMMANDS.

* :DSTAT AND :VSUSER ARE NOW USER COMMANDS ONLY.

NOTE: REPLY, ABORTIO, AND ABORTJOB MAY BE ISSUED AS COMMAND INTERPRETER OR "CNTL A" COMMANDS. THIS ALLOWS, FOR EXAMPLE, THE OPERATOR TO ISSUE A =REPLY WITHOUT HAVING TO BREAK FROM AN EXECUTING PROGRAM TO ISSUE A :REPLY. (BREAK DURING STORE AT CONSOLE ABORTS THE STORE)

SPOOLER ENHANCEMENTS

* DEVICE CLASS CAPABILITY

* NEW CONSOLE COMMANDS

SPOOLING ENHANCEMENTS

NEW CONSOLE COMMANDS

- =SPOOL, =DELETE, AND =ALTFILE HAVE BEEN REPLACED BY A NUMBER OF NEW, LESS CONFUSING CONSOLE COMMANDS

:STARTSPOOL
:STOPSPOOL

:SUSPENDSPOOL
:RESUMESPOOL

:ALTSPOOLFILE
:DELETESPOOLFILE

:STARTSPOOL

- CAUSES THE SPOOLER TO OWN A SPECIFIED DEVICE
- THE SPOOLER DETERMINES WHETHER THE DEVICE IS AN INPUT OR OUTPUT DEVICE

SYNTAX

$$:\text{STARTSPOOL} \begin{Bmatrix} \text{LDEV} \\ \text{DEVICECLASS} \end{Bmatrix}$$

<u>NOTE</u>: IF DEVICECLASS IS SPECIFIED, A DEVICE IN THE CLASS MAY BE USED UNSPOOLED BY REFERRING TO THE LDEV WHEN OPENING THE DEVICEFILE

THIS WILL BE DISCUSSED FURTHER IN THE SECTION ON DEVICE CLASS CAPABILITY

EXAMPLE: `:STARTSPOOL 6`

THE SPOOLER NOW OWNS LDEV 6

:STOPSPOOL

CAUSES THE DEVICE TO STOP AFTER PRINTING THE ACTIVE PRINT JOB (IF ANY) AND BECOME AVAILABLE.

SYNTAX

$$:\text{STOPSPOOL}\left\{\begin{matrix}\text{LDEV}\\ \text{DEVICECLASS}\end{matrix}\right\}$$

:SUSPENDSPOOL

- CAUSES THE SPOOLED DEVICE TO STOP

SYNTAX

```
:SUSPENDSPOOL LDEV [;FINISH]
```

- IF FINISH OPTION IS SPECIFIED, PRINTER WILL CONTINUE TO PRINT AND IT WILL STOP UPON COMPLETION OF THE CURRENTLY ACTIVE PRINTJOB.

:SUSPENDSPOOL 6;FINISH IS EQUIVALENT TO THE OLD

=SPOOL 6,WAIT

:RESUMESPOOL

- CAUSES THE DEVICE STOPPED BY THE SUSPENDSPOOL COMMAND TO RESUME

SYNTAX

```
:RESUMESPOOL LDEV
```

EQUIVALENT TO OLD =SPOOL LDEV,RESUME

:DELETESPOOLFILE

- DELETES THE SPECIFIED READY OR ACTIVE SPOOLFILE

SYNTAX

$$:\text{DELETESPOOLFILE} \begin{Bmatrix} \#O\text{NNN} \\ \#I\text{NNN} \\ \text{LDEV} \end{Bmatrix}$$

EXAMPLE:

TO DELETE AN OUTPUT SPOOLFILE BEING PRINTED ON LDEV 6 (LP), STOP THE PRINTER AND ENTER:

```
:DELETESPOOLFILE 6
```

:ALTSPOOLFILE

- ALLOWS USER/OPERATOR TO ALTER A READY OR LOCKED SPOOLFILE

SYNTAX

```
                            [;PRI=OUTPUTPRIORITY]
                            [;COPIES=NUMCOPIES]
    :ALTSPOOLFILE #ONNN     [;DEV= {LDEV    }]
                                   {DEVCLASS}
                            [;DEFER]
```

- THE OUTPUTPRIORITY, NUMBER OF COPIES, OR DEVICE OF SPOOLFILES MAY BE ALTERED

- THE DEFER OPTION RESULTS IN THE OUTPUTPRIORITY OF THE SPOOLFILE BEING CHANGED TO 0 IMMEDIATELY AND THE FILE BEING PLACED IN THE READY STATE.

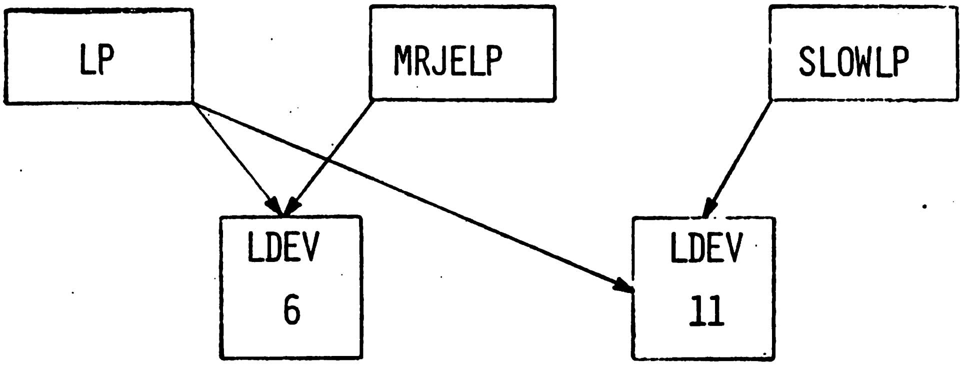
SPOOLER ENHANCEMENTS

DEVICE CLASS CAPABILITY

- ALLOWS USE OF A DEVICE AS UNSPOOLED WHILE STILL RETAINING THE ABILITY TO CREATE SPOOLED FILES FOR LATER OUTPUT TO THE DEVICE WHEN IT IS AGAIN SPOOLED

EXAMPLE:

CONSIDER TWO DEVICES AND THREE DEVICE CLASSES:



LDEV 6 IS INCLUDED IN DEVICE CLASSES "LP" AND "MRJELP"
LDEV 11 IS INCLUDED IN DEVICE CLASSES "LP" AND "SLOWLP"

ASSUME THE FOLLOWING COMMANDS ARE ISSUED:

```
:STARTSPOOL 6
:STOPSPOOL 11
```

SPOOLER ENHANCEMENTS

- DEVICE CLASS CAPABILITY EXAMPLE (CONT)

OUTPUT REFERENCED TO "LP","MRJELP", OR LDEV 6 WILL BE PRINTED VIA SPOOLFILE

OUTPUT REFERENCED TO "SLOWLP" OR LDEV 11 WILL BE PRINTED "HOT"

IF WE NOW ISSUE THE COMMANDS

```
:STOPSPOOL 6
:STARTSPOOL LP
:STOPSPOOL MRJELP
```

NOW, REFERENCES TO "MRJELP" WILL GO HOT TO LDEV 6
REFERENCES TO LDEV 6 WILL GO HOT TO LDEV 6
REFERENCES TO "LP" WILL PRODUCE SPOOLFILES WHICH WILL NOT BE PRINTED UNTIL :STARTSPOOL 6 IS ISSUED

STORE/RESTORE TAPE LABELS

* STORE TAPES MAY NOW BE LABELLED

* LOCKWORDS CAN NOW BE PLACED ON STORE TAPES

* STORE TAPES CAN NOW BE DATED (EXPIRATION DATES)

NOTE: NEW LABELLED STORE TAPES WILL NOT BE READABLE BY PRE-1918 VERSIONS OF STORE/RESTORE

PRE-1918 UNLABELLED TAPES WILL STILL BE READABLE BY RESTORE

STORE/RESTORE TAPE LABELS

EXAMPLE

```
:FILE T=T/XXX;DEV=TAPE;LABEL=BOBLAB
:STORE AAA;*T;SHOW
?9:57/#S6/16/MOUNT TAPE VOLUME BOBLAB (MAX CHARS.=?)?
=REPLY 16,7
9:57/#S6/16/VOL RES    (ANSI) MOUNTED ON LDEV# 7
THU, JUL  5, 1979,  9:57 AM

  FILES STORED = 1

    FILE      .GROUP     .ACCOUNT    LDN   ADDRESS   VOLUME

    AAA       .PUB       .STAMPS     2     %363364      1




  FILES NOT STORED = 0


:FILE T:DEV=TAPE:LABEL=BOBLAB
:RESTORE *T;AAA;KEEP;SHOW
?10:03/#S6/16/MOUNT TAPE VOLUME BOBLAB (MAX CHARS.=?)?
10:03/12/VOL BOBLAB(ANSI) MOUNTED ON LDEV# 7
TAPE LABEL LOCKWORD VIOLATION  (FSERR 121)
ERROR ACCESSING TAPE - RESTORE STOPPED.  (CIERR 1026)
:FILE T=T/XXX;DEV=TAPE;LABEL=BOBLAB
:RESTORE *T;AAA;KEEP;SHOW
?10:04/#S6/16/MOUNT TAPE VOLUME BOBLAB (MAX CHARS.=?)?
10:04/12/VOL BOBLAB(ANSI) MOUNTED ON LDEV# 7
THU, JUL  5, 1979, 10:04 AM

  FILES RESTORED = 0

  FILES NOT RESTORED = 1

    FILE      .GROUP     .ACCOUNT FILESET     REASON

    AAA       .PUB       .STAMPS      1     ALREADY EXISTS
```

STORE/RESTORE TAPE LABELS

EXAMPLE

```
:FILE T=T/XXX:DEV=TAPE;LABEL=BOBLA
:STORE AAA;*T;SHOW
?10:08/#S6/16/MOUNT TAPE VOLUME BOBLA  (MAX CHARS.=2)?
10:08/12/VOL UNLABELLED   MOUNTED ON LDEV# 7
=REPLY 16,7
10:08/#S6/16/VOL UNLABELLED   MOUNTED ON LDEV# 7
?10:08/#S6/16/MOUNT TAPE VOLUME BOBLA  (MAX CHARS.=2)?
=REPLY 16,7
10:08/#S6/16/VOL BOBLA (ANSI) MOUNTED ON LDEV# 7
?10:08/#S6/16/MOUNT TAPE VOLUME BOBLA  (MAX CHARS.=2)?
=REPLY 16,7
10:08/#S6/16/VOL BOBLA (ANSI) MOUNTED ON LDEV# 7
THU, JUL  5, 1979, 10:08 AM

  FILES STORED = 1

    FILE    .GROUP   .ACCOUNT   LDN  ADDRESS   VOLUME

    AAA     .PUB     .STAMPS    2    %363364     1

  FILES NOT STORED = 0

:RESTORE *T;AAA;KEEP;SHOW
?10:09/#S6/16/MOUNT TAPE VOLUME BOBLA  (MAX CHARS.=2)?
10:09/12/VOL BOBLA (ANSI) MOUNTED ON LDEV# 7
TAPE APPARENTLY NOT A STORE TAPE.  (CIERR 1035)
T H E R E F O R E , EXACTLY 6 CHAR. REQUIRED FOR ID
```

VIRTUAL MEMORY INCREASE

* THE LIMIT ON THE AMOUNT OF CONFIGURED VIRTUAL MEMORY ON THE SYSTEM HAS BEEN RAISED FROM 32767 SECTORS TO 65535 SECTORS.

MPE RESEGMENTATION

* AN OVERALL RESEGMENTATION OF MPE HAS RESULTED IN 9 FEWER CST ENTRIES, LEAVING MORE CST SPACE FOR USER SEGMENTS.

RESEGMENTATION OF MPE

$FIVE MODULES NO LONGER EXIST

- * Module 61 - formerly CLOCKIO
- * Module 65 - formerly PROCMAIL
- * Module 66 - formerly PINT
- * Module 68 - formerly CRIO
- * Module 77 - formerly STKDUMP

$FOUR MODULES ARE NEW

- * Module 55 - formerly DISKSPC
  - now HARDRES (CRIO & CLOCKIO combined)
- * Module 56 - formerly MMCORER
  - now SOFTRES (MMCORER & CROUTINE combined)
- * Module 58 - formerly ABORTRAP
  - now ABORTDUMP (ABORTRAP & STKDUMP combined)
- * Module 60 - formerly CROUTINE
  - now PROCSEG (PROCMAIL & PINT combined).

$ ONE MODULE HAS CHANGED

- * Module 54 - ALLOCATE
  - formerly ALLOCATE & ALLOCUTIL code segments
  - now, the code segment names are the same, but ALLOCUTIL is ALLOCUTIL & DISKSPC combined.

MPE MODULE SOURCE

SYSTEM PROGRAMS

Module Name	Mod#
INITIAL	00
SYSDUMP	01
SEGPROC	02
SEGDVR	03
DISPATCH	04
LOAD	05
UCOP	07
DEVREC	08
PROGEN	09
ININ	10
MEMLOGP	11
LOG	12
IOPTRD0	13
IOPTPN0	14
IOPLOT0	15
IOMDISC0	16
IOFDISC0	17
IOTAPE0	18
IOLPRT0	19
IOCDRD0	20
IOTERM0	22
IOPRPN0	24
IOREM0	25
IOMDISC1	27
PFAIL	30
PVPROC	31
VINIT	32

SYSTEM SL SEGMENTS

Module Name	Mod#
MAKECAT	40
FILESYS	50
COMM'INT	51
STORE/RESTORE	52
DIRC	53
ALLOCATE	54
HARDRES	55
SOFTRES	56
MMDISKR	57
ABORTDUMP	58
MESSAGE	59
PROCSEG	60
NRIO	62
PCREATE	63
MORGUE	64
DATASEG	67
CHECKER	69
UTILITY	70
SEGUTIL	71
LOADER1	72
RINS	73
JOBTABLE	74
DEBUG	75
NURSERY	76
FIRMWARESIM	78
SPOOLING	79
SPOOLCOMS	80
PVSYS	81
UDC	82
USER	83
HELPUSER	84
OPCOMMAND	85
LABSEG	86
SDISC	87
MEASIO	88
LOGSEG0	90
LOGSEG1	91

APPENDIX A

USER LOGGING

USER LOGGING PROCEDURES

```
PROCEDURE OPENLOG(INDEX,LOGID,PASS,MODE,STATUS);
DOUBLE INDEX;
INTEGER MODE,STATUS;
LOGICAL ARRAY LOGID,PASS;
OPTION EXTERNAL;
```

PARAMETERS:

INDEX
A double word returned to the user to identify the logging access. The first word of the parameter contain the logging buffer DST number, the second word contains the entry number for the user in the logging buffer. This information is used to check the validity of subsequent calls to writelog and closelog.

LOGID
An array in which the user supplies his logging identification. It can be up to eight characters long and must match a logging identifier previously established using the :GETLOG command. The logging identifier can be used to identify ownership of logging records for auditing or recovery. If the logging identifier is less than eight characters long, it must be terminated with a blank.

PASS
An array in which the user supplies the pass word associated with the logging identifier. To obtain access, the password must match the password established with the logging identifier using the :GETLOG command. The password can be up to eight characters long. If the password is less than eight characters, it must be terminated with a blank.

MODE
An integer in which the user indicates whether or not his process should be suspended in the event that his access to the logging facility cannot be obtained immediately. This can occur when the user is sharing a logging buffer which is currently full and the logging process has been too busy to empty it. If the user wants to wait for his access, he should enter a zero in the mode parameter, if the user does not want to wait, he should enter one.

STATUS
An integer that the logging system uses to return status information to the user.

DESCRIPTION:

USER LOGGING PROCEDURES

The OPENLOG intrinsic is used to obtain accesss to a user logging file. The intrinsic insures that the caller has access to the logging file by checking the validity of the logging identifier and pass word. If the access is found to be valid, a double word is passed back to the user to be used in calls to writelog or closelog. A record is also written in the User Logging File that identifies the access to the file.

USER LOGGING PROCEDURES

```
PROCEDURE WRITELOG(INDEX,DATA,LEN,MODE,STATUS);
DOUBLE INDEX;
INTEGER LEN,MODE,STATUS;
LOGICAL ARRAY DATA;
OPTION EXTERNAL;
```

PARAMETERS:

INDEX
The parameter returned from OPENLOG that identifies the user's access to the logging file. For privileged users, the first word of the index parameter can be used to pass a subsystem code. This code is placed in the top half of the code word on the logging record and can be used by subsystem or privileged user recovery systems.

DATA
An array in which the user passes the actural information to be logged. A log record contains 128 words of which 119 is available to the user. .The most efficient use of log file space is a multiple of 119 words.

LEN
The length of the data in DATA. A positive count indicates words a negative count indicates bytes. If the length is greater than 119 words, the information in data will be divided into two or more physical log records.

MODE
An integer in which the user indicates whether or not his process should be suspended in the event that his access to the logging facility cannot be obtained immediately. This can occur when the user is sharing a logging buffer which is currently full and the logging process has been too busy to empty it. If the user wants to wait for his access, he should enter a zero in the mode parameter, If the user does not want to wait, he should enter one.

STATUS
An integer that the logging system uses to return error information to the user.

USER LOGGING PROCEDURES

DESCRIPTION:

The WRITELOG intrinsic is used to write a physical record to a logging file. When the user calls the intrinsic it, provided there is room, places the request into the buffer area of the logging data segment. If the request fills the buffer, the intrinsic will write the contents of the buffer to the disc logging buffer for a tape log file, or the user's disc logging file for a disc log file. After writing the buffer, the intrinsic will awake the logging process using the PCB number stored in the communications area of the logging data segment.

In the event that the disc buffer used for logging to tape becomes full, the intrinsic will sleep the caller unless a nowait mode has been requested in which case the intrinsic will return to the caller without processing the request. In this case, it is up to the caller to re-issue the request at a later time.

USER LOGGING PROCEDURES

```
PROCEDURE CLOSELOG(INDEX,MODE,STATUS);
DOUBLE INDEX;
INTEGER MODE,STATUS;
OPTION EXTERNAL;
```

PARAMETERS:

INDEX
The parameter returned from OPENLOG that identifies the user's access to the logging file.

MODE
An integer in which the user indicates whether or not his process should be suspended in the event that his access to the logging facility cannot be obtained immediately. This can occur when the user is sharing a logging buffer which is currently full and the logging process has been too busy to empty it. If the user wants to wait for his access, he should enter a zero in the mode parameter, if the user does not want to wait, he should enter one.

STATUS
An integer that the logging system uses to return error information to the user.

DESCRIPTION:

The CLOSELOG intrinsic is used to close a user log file. When the intrinsic is called, the logging entry in the logging data segment is deleted and any information remaining in the buffer portion of the logging data segment is flushed to the destination file. A record identifying the close of the log file is also written.

QUIZ - A GENERALIZED REPORT WRITER FOR THE HP 3000

by: Robert Minns and Fred H. Smith
Quasar Systems Ltd.
99 Bank Street
Suite 641
Ottawa, Ontario
Canada
(613) 237-1440

ABSTRACT

QUIZ is a generalized reporting and inquiry program for the HP 3000. Simple ad-hoc reports can be produced by an end-user through entry of minimal specifications. Complex reports may be produced by entry of detailed format specifications. QUIZ operates on any combination of IMAGE/3000 databases, KSAM/3000 files and conventional MPE files.

This paper reviews our experience in the design and implementation of QUIZ. Particular emphasis is placed upon the difficulties of designing a facility which is simple enough to be used by non-EDP personnel to produce simple reports, but which is also capable of the sophisticated formatting and control required of production reports. User experience in a number of installations is also described.

INTRODUCTION

QUIZ is a generalized report writer for the HP 3000. QUIZ was originally developed as an internal tool for use by Quasar Systems Ltd. staff in the implementation of turnkey business systems based on the HP 3000. However, the enthusiastic response of our customers has encouraged us to further develop QUIZ into a generalized reporting package.

Quasar was led to develop QUIZ because of a need to provide an inquiry facility for MPE and KSAM files and because of the unsuitability of QUERY as a production report writer for IMAGE databases. Our primary design objective was to develop a tool which could be used both for simple inquiries and ad-hoc reports by clerical level staff and which could also be used by programmers to generate complex production reports.

This paper reviews our experience in the design and implementation of QUIZ. Particular emphasis is placed on the difficulty in meeting the primary design objective.

DESIGN OBJECTIVES

As indicated in the Introduction, the primary objective of the design of QUIZ was to develop a tool which could act both as a database inquiry facility and as a production report writer. This objective can never be completely realized, since the needs of the groups which perform each function are radically different. Therefore, the primary objective was translated into a set of more realistic sub-objectives, which are listed below:

- the report writer must support any combination of MPE, KSAM and IMAGE files, including multiple IMAGE databases in a single report.

- the report writer must allow unlimited logical linkages between keyed files, independent of the physical representation of the data.

- the report writer must be able to produce properly formatted reports from minimal specifications, but it must also be possible to override the automatic formatting and specify the format of the report in detail.

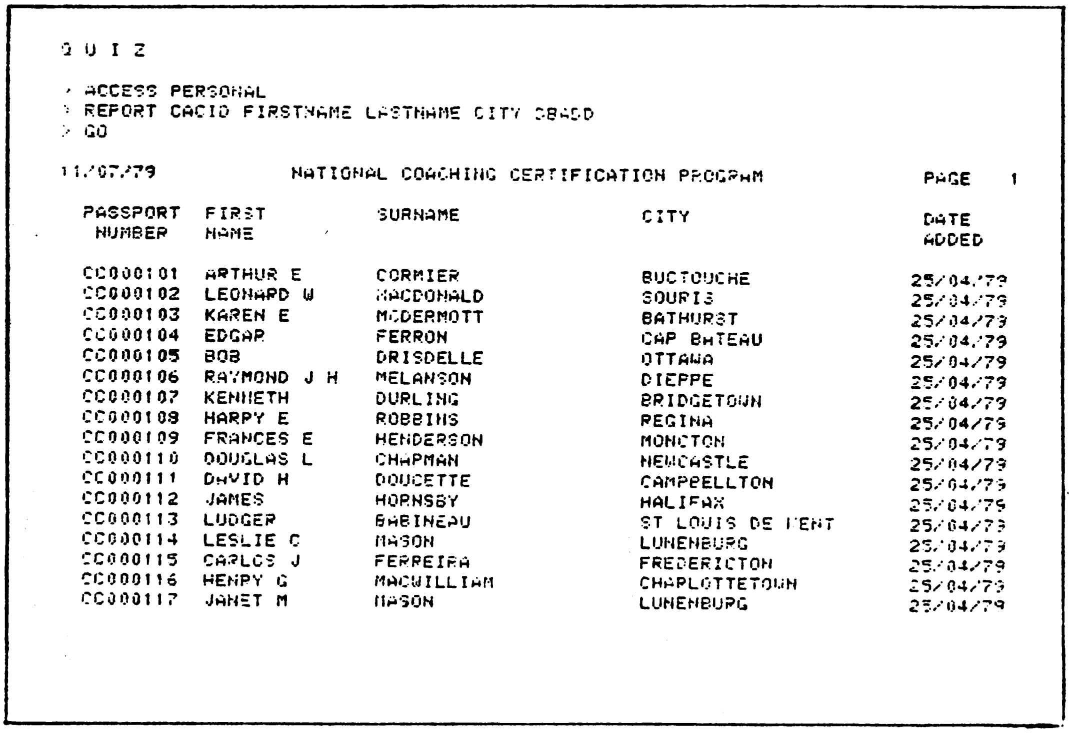
- the report writer must provide the formatting capabilities required by commercial applications, including leading and/or trailing signs, credit/debit indication, currency symbol, punctuation of numeric fields, blank when zero, "n-up" reporting (e.g. 4-up mailing labels) etc..

- the report writer must provide an extensive run-time calculation capability. This capability must include numeric, string and date expressions and the ability to produce counts and subtotals in control breaks.

These objectives are, of course, additional to the basic objectives which should apply to all software development (packaged or otherwise), i.e. that the programs should be easy to use, reasonably efficient, well documented, modular, easily modifiable, etc..

MEETING THE OBJECTIVES

The report specification and a sample page of output from a simple ad-hoc QUIZ report are reproduced in figure 1 below.



```
Q U I Z

> ACCESS PERSONAL
> REPORT CACID FIRSTNAME LASTNAME CITY DBADD
> GO

11/07/79            NATIONAL COACHING CERTIFICATION PROGRAM                  PAGE   1

  PASSPORT  FIRST         SURNAME                  CITY                      DATE
   NUMBER   NAME                                                             ADDED

  CC000101  ARTHUR E      CORMIER                  BUCTOUCHE                25/04/79
  CC000102  LEONARD W     MACDONALD                SOURIS                   25/04/79
  CC000103  KAREN E       MCDERMOTT                BATHURST                 25/04/79
  CC000104  EDGAR         FERRON                   CAP BATEAU               25/04/79
  CC000105  BOB           DRISDELLE                OTTAWA                   25/04/79
  CC000106  RAYMOND J H   MELANSON                 DIEPPE                   25/04/79
  CC000107  KENNETH       DURLING                  BRIDGETOWN               25/04/79
  CC000108  HARRY E       ROBBINS                  REGINA                   25/04/79
  CC000109  FRANCES E     HENDERSON                MONCTON                  25/04/79
  CC000110  DOUGLAS L     CHAPMAN                  NEWCASTLE                25/04/79
  CC000111  DAVID H       DOUCETTE                 CAMPBELLTON              25/04/79
  CC000112  JAMES         HORNSBY                  HALIFAX                  25/04/79
  CC000113  LUDGER        BABINEAU                 ST LOUIS DE KENT         25/04/79
  CC000114  LESLIE C      MASON                    LUNENBURG                25/04/79
  CC000115  CARLOS J      FERREIRA                 FREDERICTON              25/04/79
  CC000116  HENRY G       MACWILLIAM               CHARLOTTETOWN            25/04/79
  CC000117  JANET M       MASON                    LUNENBURG                25/04/79
```

FIGURE 1: SIMPLE AD-HOC REPORT

Three commands were required to produce the report. The ACCESS command identifies the file to be read. The REPORT command identifies the data items to be reported. The GO command initiates report production. This sequence, possibly augmented by a SELECT command, is typical of what is required to process online inquiries or simple ad-hoc reports.

The simplicity of this report specification is accomplished in two ways. First, QUIZ provides default formatting specifications which supply a standardized page heading including the current date, the application name, the current page number and field headings. A simple algorithm is used to space the data items across the page and to justify or centre field headings.

Second, the detailed format instructions for each data item have already been provided by the database administrator or system designer. In this sample report, the application name, the field headings and the special formatting applied to the items CACID and DBADD were obtained from a file schema prepared by the database administrator and accessed by QUIZ at run time. The file schema is a description of an application "database" (which may include several (or zero) IMAGE databases, as well as KSAM or MPE files). The description identifies the qualified users of the database, the files which make up the database, the data elements contained in the files and the structure and contents of each record. The description may contain file and element security specifications, data element headings, data element scaling and output format options. The database description need be prepared only once - it is then available to all users of the database.

The previous example illustrates how we have attempted to meet the needs of the unsophisticated user of QUIZ. The next example illustrates how the more sophisticated user can exercise considerable control over the output of QUIZ. The example is also intended to illustrate the facilities provided to meet the design objectives listed in a previous section.

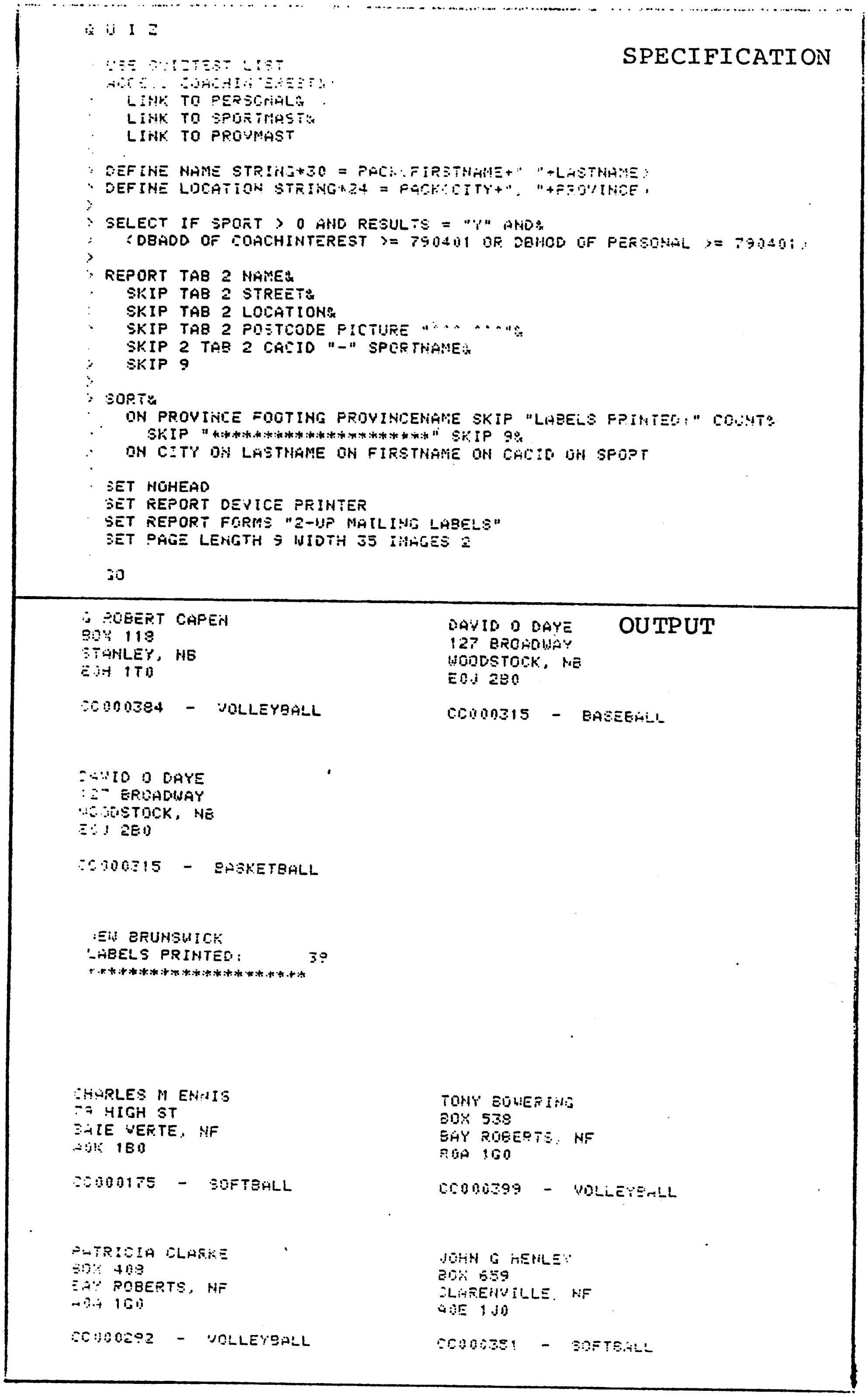
Figure 2 contains the report specifications and sample output of a QUIZ run to produce mailing labels. The example illustrates multi-file access, run-time definition of temporary variables, selection of data, user specified formatting, sorting and control breaks and parameter setting.

The ACCESS command in this example directs QUIZ to read each record of the COACHINTEREST file (an IMAGE detail file containing a list of sports coached), to read the corresponding PERSONAL record (another IMAGE detail file containing name and address information) and to read sport name and province name from the appropriate master files. In this example, the linkages between the files are established automatically by QUIZ which searches for an occurrence of the primary key item of newly accessed files in one of the preceding files. For secondary key access, or for linkages between items with different names, the linkage may be explicitly specified.

The DEFINE commands illustrate the definition of run-time character string variables. NAME is defined to be a 30 character string made up of a concatenation of the first name, a space and the last name, packed together so that leading and all but one intervening spaces are suppressed. Numeric and date expressions may be defined in a similar manner.

The SELECT command defines a selection criterion applied to each (compound) record read. Note that parentheses may be used to alter the precedence of the AND and OR connectors.

The REPORT command specifies the contents and format of the detail report lines. TAB and SKIP parameters are used to specify the format of the detail report group. TAB specifies the column in which the printing of a field should begin. SKIP indicates the number of lines to be skipped before the next field is printed. The default format of any data item may also be overridden. In the example, the POSTCODE field is to be printed with an additional space between the initial three and final three characters.



SPECIFICATION
OUTPUT

```
QUIZ

USE QUIZTEST LIST
ACCESS COACHINTEREST&
   LINK TO PERSONAL&
   LINK TO SPORTMAST&
   LINK TO PROVMAST

DEFINE NAME STRING*30 = PACK(FIRSTNAME+" "+LASTNAME)
DEFINE LOCATION STRING*24 = PACK(CITY+", "+PROVINCE)

SELECT IF SPORT > 0 AND RESULTS = "Y" AND&
   (DBADD OF COACHINTEREST >= 790401 OR DBMOD OF PERSONAL >= 790401)

REPORT TAB 2 NAME&
   SKIP TAB 2 STREET&
   SKIP TAB 2 LOCATION&
   SKIP TAB 2 POSTCODE PICTURE "^^^ ^^^"&
   SKIP 2 TAB 2 CACID "-" SPORTNAME&
   SKIP 9

SORT&
   ON PROVINCE FOOTING PROVINCENAME SKIP "LABELS PRINTED:" COUNT&
      SKIP "*********************" SKIP 9&
   ON CITY ON LASTNAME ON FIRSTNAME ON CACID ON SPORT

SET NOHEAD
SET REPORT DEVICE PRINTER
SET REPORT FORMS "2-UP MAILING LABELS"
SET PAGE LENGTH 9 WIDTH 35 IMAGES 2

GO
```

```
G ROBERT CAPEN                    DAVID O DAYE
BOX 118                           127 BROADWAY
STANLEY, NB                       WOODSTOCK, NB
E0H 1T0                           E0J 2B0

CC000384  -  VOLLEYBALL           CC000315  -  BASEBALL


DAVID O DAYE
127 BROADWAY
WOODSTOCK, NB
E0J 2B0

CC000315  -  BASKETBALL


NEW BRUNSWICK
LABELS PRINTED:      39
*********************



CHARLES M ENNIS                   TONY BOWERING
79 HIGH ST                        BOX 538
BAIE VERTE, NF                    BAY ROBERTS, NF
A0K 1B0                           R0A 1G0

CC000175  -  SOFTBALL             CC000399  -  VOLLEYBALL


PATRICIA CLARKE                   JOHN G HENLEY
BOX 408                           BOX 659
BAY ROBERTS, NF                   CLARENVILLE, NF
A0A 1G0                           A0E 1J0

CC000292  -  VOLLEYBALL           CC000351  -  SOFTBALL
```

FIGURE 2: A MORE COMPLEX REPORT

The SORT command specifies the order in which the selected data is to be reported and identifies any control breaks required. Six sort keys are used in this example, specified in sequence from the major key (PROVINCE) to the most minor key (SPORT). A control break will occur on PROVINCE, and a control footing (summary) will be printed including the name of the province, a count of the number of labels printed for the province and a line of asterisks. The syntax of the specification of control headings or footings is identical to the syntax of the REPORT statement. The same syntax may also be used to specify an initial heading, a final footing, page headings and page footings.

The mailing label report also uses several parameter settings. The standard page heading is eliminated (NOHEAD). Output is routed to the system printer and a forms message is entered for display to the operator. The output page is set to a standard mailing size (9 lines x 35 characters, "2-up").

PRACTICAL EXPERIENCE

The reaction of our clients to QUIZ has been very enthusiastic. However, we have learned the truth of the maxim "you can't please everybody". In particular, one cannot expect to provide all the features required by programmers involved in the production of complex reports or processing complex file structures. "One more" feature is always required. As these additional features are added, the complexity of the program and, especially, its documentation and help facilities increases and it becomes less attractive as a vehicle for simple database inquiry. It has proved to be relatively straightforward to add additional formatting options to QUIZ, without obscuring the simplicity of the design. We have been much less successful in accommodating more complex file structures, such as header and detail records within the same file, multiple record types from the same file reported simultaneously, and complex variable length structures.

QUIZ has now been installed on five HP 3000 computers across Canada. A variety of application systems have been implemented using QUIZ both as an inquiry tool and for generation of production reports. Applications have included a licensing system, a sports result system, a payroll system and a number of standard business applications. QUIZ has been used to print paycheques, membership cards, invoices, mailing labels and other forms, in addition to standard tabular reports. We are confident that the use of QUIZ in these applications has resulted in substantial reductions in cost of development of production reports and inquiry facilities.

DESIGN AND IMPLEMENTATION OF REX/3000:
A NATURAL LANGUAGE REPORT WRITER

LANCE CARNES
GENTRY, INC
KENSINGTON, CALIFORNIA

This paper presents a natural language report writer, REX/3000 (Report EXpediter), designed for the casual user as well as the experienced programmer. The language design and compiler implementation is discussed from the point of view of meeting the needs of the two user classes. For the casual user, the language is natural and English-like. For the programmer, the language is a mixture of natural and procedural constructs. The PASCAL-based compiler has been implemented to serve each user class appropriately.

INTRODUCTION.

REX was developed to facilitate writing reports from IMAGE databases and MPE files. The user will range from the casual (infrequent) non-programmer attempting to formulate reports to the experienced programmer with more complex goals.

The time allotted for the design and implementation was eight man-months. The project consisted of the following:

1) Design the syntax of REX, i.e. the form of the language.

2) Design the semantics of REX, i.e. the function of the compiled programs, to properly execute for both the casual user and the programmer.

3) Write a compiler for REX.

This seemingly tight schedule was supported by several resources:

1) The author of a report writer similar to the required package (though unfortunately written in IBM 360 assembler language [1]) was available for consultation;

2) A PASCAL compiler had been transported to the HP by a user, and graciously contributed to the user's library [2] ;

3) A copy of the production PASCAL-P compiler source (written in PASCAL) had also been contributed to the users library.

This paper will treat these topics, discussing the design considerations, and the solutions implemented.

SYNTAX.

The careful design of the syntax of a language is important. From the compiler writer's point of view, the syntax should be simple and have as few rules and exceptions as possible. From the language user's point of view, especially the casual user, the syntax should be natural and English-like as well as consistent and unambiguous.

The syntax of a language is the way the elements of the language are put together to form statements. For example, in English many statements have the syntax

```
    subject  verb  object .
```

In computer languages the syntax of assignment statements is commonly:

```
    variable  =  number
 or
    variable  =  number + number
 or, in the general case,
    variable  =  expression
```

The syntax of REX lies somewhere between a typical programming language and a natural language. The total logical precision of a programming language is not wanted, since the casual user is not trained in the use of these languages. Nor do we want the full complicated syntax of English, since the language must be translated by a computer program (the compiler).

The syntax of REX is modeled after the syntax of many of the currently used 'natural language' report writers: SYNTAX II [1], QUERY [3], NATURAL [4], R11 [5], and RAMIS II [6]. The intent of all the natural language report writers is to avoid the 'procedural' or programming details of programming languages and to allow the user to formulate English-like specifications which can be interpreted by a compiler.

Consider Example 1, a REX program which reads and prints selected values from a database called PAYROLL.

```
<< LIST EMPLOYEES IN SALES DEPARTMENT FROM MARIN COUNTY >>
database PAYROLL password "READER" access 8
  dataset EMPLOYEES
print "SALES PERSONNEL RESIDING IN MARIN COUNTY"
set EMPLOYEES with DEPT = "SALES"
  select ZIP isbetween 94100,94199
  print EMPL-NAME, EMPL-ADDR, EMPL-NUM
end.
```

Example 1. A complete REX program.
(Words in lowercase are keywords - in actual practice upper and lower case letters are treated the same.)

This is a complete REX program and will produce a listing of the desired values. (Its actual function will be discussed in the next section, "Semantics". See Appendix B for sample programs with printed output.)

The point here is that REX has an English-like syntax which serves the casual user. Notice that there are no procedural steps (i.e. there is no explicit opening or closing of the database, no explicit test for end-of-chain, etc). In a sense, the syntax closely resembles the specification of the program.

For the experienced programmer, who must deal with more complex programming requests, there are more complete constructs available. However, these constructs are not always "natural language" in nature and would not be used commonly by the casual user. The constructs are available for completeness and are not in ordinary demand by the non-programmer.

For example, suppose the user wishes to read an MPE file containing employee numbers, look up each employee number in the PAYROLL database, and write a new file with employee name, address and number.

```
<< READ EMPLOYEE #, LOOK UP # IN PAYROLL DATABASE,
<< WRITE NAME, ADDRESS, EMPLOYEE #

file EMP#FILE = 'EMPLNO.PUB.ACCOUNTS' f10  << INPUT FILE >>
  EMP# 1-4 n7  << 4-BYTE FIELD, 7-DIGIT INTEGER >>

database PAYROLL password 'READER' access 8
  dataset EMPLOYEES

file EMP-NAME-FILE = 'EMPLNAMS.PUB' f72 << OUTPUT FILE >>
  E-NAME  1-20 a20   << 20-BYTE ALPHA FIELD >>
  E-ADDR 21-50 a30
  E-NUM  51-54 n7    << 7-DIGIT INTEGER >>

progvar REC-CTR   n8 << 8-DIGIT COUNTER, RECORDS READ >>
        NOT-FOUND n8 << 8-DIGIT COUNTER, INVALID EMPL #'S >>

begin  << MAIN PROGRAM >>

  REC-CTR = REC-CTR + 1  << COUNT INPUT RECORDS >>

  get first EMPLOYEES with EMPL-NUM = EMP#  << LOOK UP >> &
    at end begin
             NOT-FOUND = NOT-FOUND + 1 << LOOK UP FAILED >>
             print 'EMPL # ' EMP# ' NOT IN PAYROLL'
           end
  E-NAME = EMPL-NAME   << MOVE VALUES TO OUTPUT RECORD >>
  E-ADDR = EMPL-ADDR
  E-NUM  = EMPL-NUM
  write EMP-NAME-FILE    << WRITE OUTPUT RECORD >>
loop  << DO NEXT read EMP#FILE >>

<< ALL RECORDS PROCESSED, PRINT TOTALS >>
if NOT-FOUND <> 0 then  &
  print 'TOTAL EMPL #'S NOT FOUND = ' NOT-FOUND  &
else print 'ALL EMPL #'S FOUND
print 'TOTAL RECORDS PROCESSED = ' REC-CTR
end.
```

Example 2. A complex program.

This example illustrates many of the constructs available to the programmer or the adventuresome casual user. See Appendix A for a summary of the REX language.

SEMANTICS.

The syntax of a language provides a framework of correctly formatted statements which may then be converted to meaningful, functioning programs. The conversion of statements into functional programs is called the semantics of the programming language. The semantics for REX are such that the constructs used by the casual user require a great deal of implicit function, while the procedural constructs for the experienced programmer are explicit.

For example, in English the request

"Look at your watch and recite the exact hour and minute."

may also be stated as

"Tell me the time."

The first form is correct English syntax and there is no difficulty understanding what is desired, although this is not the usual way to ask the time. The second form has the same meaning and can be understood with slightly more effort; i.e. you must subconsciously recall that you need to look at your watch, and recite the hour and minute. The first form is procedural and explicit; the second form is natural and requires a great deal of implicit function.

The semantics of REX are designed to cover both cases - the casual user can express a natural request while the programmer can code the same function procedurally. To illustrate, recall Example 1. Nowhere was it indicated explicitly that the dataset was to be read using a chained read on the DEPT search item, looping back to the 'get EMPLOYEES' statement after each entry had been printed. The explicit statement of this program is shown in Example 3.

```
database PAYROLL
  dataset EMPLOYEES
print 'SALES PERSONNEL RESIDING IN MARIN COUNTY'
find EMPLOYEES with DEPT = 'SALES'   << FIND CHAIN HEAD >>
get next EMPLOYEES at end stop   << NEXT ENTRY ON CHAIN >>
  select ZIP isbetween 94100,94199
  print EMPL-NAME, EMPL-ADDR, EMPL-NUM
loop    << END OF get LOOP >>
end.
```

Example 3. Explicit statement of Example 1.

The program in Example 3 will produce the exact same result as the program in Example 1. Details are described which were filled in by the compiler in Example 1. The clause 'at end stop' indicates that processing is to cease at the end-of-chain. The 'loop' statement indicates that the processing loop ends precisely here, so that statements may be written following the loop which are not executed until the loop is terminated (see Example 2).

In all aspects of REX, the functions which are most used by the casual user have a "natural language" or non-procedural default. Another example of this aspect of REX is the report block. The following program produces a result similar to the program in Example 1 - except here the output is sorted by employee name, and a report title and column headings are provided:

```
database PAYROLL
  dataset EMPLOYEES
report "SALES PERSONNEL RESIDING IN MARIN COUNTY"
  select ZIP isbetween 94100,94199
  list EMPL-NAME, EMPL-ADDR, EMPL-NUM  &
    sorted by EMPL-NAME
end
set EMPLOYEES with DEPT = "SALES"
end.
```

Example 4. A sorted report.

Here we have specified the contents of a report (report...end) and the method of obtaining items for the report

(set EMPLOYEES ...). What is left out of the specification (but included in the implicit function) is the linkage between the "report" and the "set", and the indication of when to sort the data and print the report.

Consider the explicit version of this same program:

```
database PAYROLL
  dataset EMPLOYEES
report MARIN-RES 'SALES PERSONNEL RESIDING IN MARIN COUNTY'
  select ZIP isbetween 94100,94199
  list EMPL-NAME, EMPL-ADDR, EMPL-NUM  &
    sorted by EMPL-NAME
end
set EMPLOYEES with DEPT = 'SALES'
  call MARIN-RES   << UPDATE REPORT DATA >>
loop
print MARIN-RES    << SORT AND PRINT THE REPORT >>
end.
```

Example 5. Explicit statement of Example 4.

Notice that the report block now has an identifier 'MARIN-RES' which allows it to be 'call'-ed, much as a procedure block. The statement 'print MARIN-RES' explicitly specifies the exact point at which the report is output.

Thus the semantics of REX are intended to appeal to both the casual and programmer user. The programmer is not saddled with a restrictive language set and may tackle more complicated programs; the casual user may ignore the procedural aspect of REX or may, with some assistance or experimentation, improve his results by venturing into the procedural constructs of the language. Acceptable results may be achieved from either level of expertise.

THE COMPILER.

One of the original goals underlying the implementation of REX was the low-cost development of a compiler which could be transported to other machines. Most of this goal was achieved. The compiler was developed at low cost (eight man-months) and is transportable (written in PASCAL). However, since the object code produced by the compiler is SPL, the package will run only on the HP at present.

A compiler is a program which processes source text in a pre-defined language to produces some form of object code. The syntax and semantics of the language to be compiled can greatly affect the complexity, and therefore the cost, of developing and maintaining a compiler. When aiming at low-cost development, it is important to keep it as simple as possible without sacrificing the utility of the language.

REX is not an easy language to compile. However, the task was reduced because a production compiler was used as a model (the PASCAL-P compiler written in PASCAL). Also, some language design decisions eased the translation effort without reducing the capability of the language.

As an example, the element <expression> was used everywhere that it made sense. This allows any production in the language which uses <expression> to share a common compiling procedure. Of course, if the statement needs an expression of a certain type, say logical in the case of "if <expression>", the compiler will flag an expression of any other type as an error.

In contrast, the language SYNTAX II [1] has several expression types which are compiled distinctly, depending on the expected type. In the case of the assignment statement SYNTAX II has several different syntactic forms depending on the destination data type.

1	A EQ B	A,B CHARACTER TYPE
2	SET A = B + 1 or A = B + 1	A,B NUMERIC TYPE
3	RECODE A TO B	A,B ANY SAME TYPE
4	T TEST A < B	T LOGICAL TYPE

Figure 1. Sample assignment statements from SYNTAX II.

The result is that there are several additional compiling procedures that must be developed and maintained.

Using a construct wherever it makes sense is desirable from the compiler writer's point of view and also from the user's point of view [7] [8]. When the user must formulate his idea differently for each data type used, the language designer has burdened the user with additional effort. For example, in the SYNTAX II "SET" statement it is legal to write

```
SET A = B * 100
```

while in the "TEST" statement it is not legal to write

```
T TEST A < (B * 100)
```

even though A and (B * 100) are of the same type. The user must remember which operators can be used in which contexts.

This requires an additional level of expertise that we wish to avoid. However, if there is only one set of rules for the formation of expressions, the user need not recall a lot of exceptions to the rules.

The REX compiler was implemented in PASCAL, a high-level recursive language [9]. The production PASCAL-P compiler source code [10] was used as the basis and as a model for writing the REX compiler. SPL was chosen for the compiler's object code primarily because it is the language closest to the machine level. SPL compiles to produce the most resource-efficient run-time programs. It runs with the least memory usage and fastest execution time compared to other compiled object code (FORTRAN or COBOL). Output in relocatable code was not chosen since there is no documentation currently available from HP on its format.

The appearance of the compiler from the user's standpoint is much like any other HP compiler. There are UDC's for invoking the compiler which resemble the commands for invoking the SPL, COBOL or FORTRAN compilers. Error messages are clear and point to the place in the text where the error occurred:

```
   select ZIP isbetwee 94100,94199
                     ^
   ****  UNKNOWN SYMBOL     (GCPERR 72)
```

Should the user want more information, he can refer to the user's manual under "GCPERR 72".

CONCLUSIONS.

REX has many powerful features which will allow the user to generate accurate, timely, and complete reports with a minimal program development effort. Results with comparable packages show a decrease in development time by as much as a factor of ten [6], compared with developing the same application in COBOL or FORTRAN.

The casual user can benefit greatly from bypassing the data processing department and writing his own ad hoc reports. The programmer can increase his productivity by reducing the development effort required to produce routine reports and queries.

At the time this paper was written, REX had been implemented with all features except the cross-tabulation, which will be completed by the time this paper is presented. There had been no casual user experience with REX, but by the time of presentation there should be real feedback as to its effectiveness with this user class.

There is no data at present regarding acceptance by experienced programmers. Due to its close resemblance to the 'ALGOL-like' languages, it is expected to be successful with programmers with this type of experience. Data will be forthcoming as to its acceptance by programmers experienced in COBOL, FORTRAN, BASIC and other languages.

Future extensions include full IMAGE access (DBPUT, DBDELETE, DBUPDATE), and full KSAM access.

REX will be marketed by

GENTRY INC.
609 Kearney Street
Kensington, California 94530
U.S.A.

The first release of REX will be available January 1980.

ACKNOWLEDGEMENTS.

My thanks to John Fitz, Richard Gentry, and Ron Frankel for many valuable discussions during the language definition phase of the project; and my special thanks to Grace Gentry, who provided me the opportunity to do this project and who assisted in the preparation of this paper.

REFERENCES.

1. John Fitz, SYNTAX II USER'S GUIDE. University of California, 1977.
2. Bob Fraley, PASCALP, HP User's library.
3. QUERY. Hewlett-Packard Product # 32216A.
4. NATURAL Users Manual, Software AG, 1978.
5. RII, Computer Sciences Corp.
6. RAMIS II, Mathematica Corp.
7. G. Weinberg, The Psychology of Computer Programming, Van Nostrand Reinhold, 1971.
8. W.M. McKeeman, "Programming Language Design" from Compiler Construction, F.L.Bauer,ed., Springer-Verlag, 1976.
9. K. Jensen and N. Wirth, PASCAL User Manual and Report, Springer-Verlag, 1974.
10. Urs Ammann, PASCAL P4 compiler source code, Zurich, 1976.

Appendix A: Summary of REX/3000 Language.

```
REPORT "TITLE"                                 Sorted report.
  LIST ...
    SORTED BY ...
END

TABLE "TITLE"                                  Cross-tabulation.
  ROW ...
  COLUMN ...
END

PROCEDURE name                                 Procedure.
  ...
END

DATABASE basename                              IMAGE declaration.
  DATASET setname

FILE filename                                  MPE file declaration.
  field1 ...
  field2 ...
  ...

PROGVAR var1 ...                               Variable declaration.
        var2 ...
        ...

                                               Database access.

GET   setname [WITH searchitem = expression]
FIND          [AT END statement]

READ  filename [AT END statement]

WRITE filename                                 File access.

PRINT expression [, expression, ...]           Unformatted print.

                                               Sort/Merge.
SORT filename [INTO filename]
         BY key1, key2, ...

MERGE filename, filename [, filename, ...]
         INTO filename
         BY key1, key2, ...
```

```
IF expression THEN statement                 Control statements.
             [ELSE statement]

GOTO label

CALL blockname

REPEAT statement [; statement; ...] UNTIL expression

WHILE expression DO statement

FOR ident = expression TO expression DO statement


BEGIN [statement; statement; ...] END   Compound statement.

ident = expression                      Assignment statement.


+ - * / DIV MOD                         Arithmetic operators.

GT GE LT LE EQ NE ISBETWEEN CONTAINS    Relational operators.
>  >= <  <= =  <> IB        <*>

AND OR NOT                              Logical operators.

                                        Data types.
Nw    w <= 10        Integer
Fw.d  d <= w <= 15   Real
L                    Logical
Aw    w <= 256       Alpha
```

APPENDIX B: SAMPLE REX/3000 REPORTS.

The following are several reports generated by REX/3000 from MPE files and IMAGE databases.

Sample 1. Sorted report using MPE file.

```
FILE PARTS: F80
   PN 'PART NO' 1-4
   PD 'PART DESC' 5-14
   PL 'LOCATION' 15-17
   QTY 'QUANTITY' 18-22 N5
REPORT 'WAREHOUSE PARTS SUMMARY'
   LIST PN, 5X, PL, 2X, PD, 2X, QTY          &
      SORTED BY PN, PL                       &
      SUMMARIZING QTY ON PN                  &
      TOTALING 'TOTAL' QTY
END
READ PARTS  << READ FILE RECORD >>
END.
```

```
1785BOLT 1 X 1/4 101 2000
2142BRACKET      100  750
3122MANUAL #177  101  100
2142BRACKET      102  250
2142BRACKET      101  100
1785BOLT 1 X 1/4 100 1000
```

Contents of the PARTS file.

The REX/3000 report:

```
               WAREHOUSE PARTS SUMMARY

PART NO      LOCATION     PART DESC       QUANTITY

1785         100          BOLT 1 X 1/4        1000
             101          BOLT 1 X 1/4        2000
-------                                   --------
1785                                          3000

2142         100          BRACKET              750
             101          BRACKET              100
             102          BRACKET              250
-------                                   --------
2142                                          1100

3122         101          MANUAL #177          100
-------                                   --------
3122                                           100
                                          --------
TOTAL                                         3200
```

Sample 2. Sorted report with IMAGE/3000 database.

The following is an IMAGE/3000 schema for a database to hold the same information from Sample 1:

```
BEGIN DATABASE WAREHOUSE;

ITEMS:
 PART-NO,     X4;
 PART-DESC,   X10;
 PART-LOC,    X4;
 PART-QTY,    I2;

SETS:

 NAME: PARTS, DETAIL;
 ENTRY:
   PART-NO,
   PART-DESC,
   PART-LOC,
   PART-QTY;
 CAPACITY: 100;

...

END.
```

The following REX/3000 report specification will produce the same report as in Sample 1:

```
DATABASE WAREHOUSE
  DATASET PARTS

REPORT "WAREHOUSE PARTS SUMMARY"
   LIST PART-NO, 5X, PART-LOC, 2X, PART-DESC, 2X, PART-QTY N5    &
      SORTED BY PART-NO, PART-LOC                                 &
      SUMMARIZING PART-QTY ON PART-NO                             &
      TOTALING "TOTAL" QTY
END
GET PARTS  << READ DATASET >>
END.
```

Sample 3. Cross-tabulation with IMAGE/3000 database.

This sample uses the same database as in Sample 2.

```
DATABASE WAREHOUSE
  DATASET PARTS

TABLE "PARTS DISTRIBUTION"
  ROW PART-NO BINS("1785","2142","3122")
  COLUMN PART-LOC BINS(100,101,102) ACCUMULATE PART-QTY
         TOTAL-PARTS LABEL "TOTAL PARTS" ACCUMULATE PART-QTY
END
GET PARTS  << READ FROM DATABASE >>
END.
```

The following cross-tabulation will be produced:

```
           PARTS DISTRIBUTION

                PART-LOC
PART-NO      100   101   102    TOTAL PARTS

1785        1000  2000     0           3000

2142         750   100   250           1100

3122           0   100     0            100
```

DOLLAR-FLOW: FINANCIAL PLANNING ON THE HP3000
(users write their own programs)

By Jack Damm, Principal, The Palo Alto Group, Sunnyvale, Calif. (408) 735-8490

Good afternoon. I am going to talk about financial planning on the HP3000 with the Dollar-Flow planning language. My discussion will focus on three areas: 1) What financial planning is, and why there is a need for computerized planning; 2) Design considerations for "friendly" user-oriented applications; and 3) How the language Dollar-Flow is used for applications like profit planning.

THE NEED FOR FINANCIAL PLANNING

First, let's start with two questions: What is financial planning? And why is it necessary? Financial planning is making decisions about allocating the scarce resources of an organization so as to best achieve its goals. In the private sector, this usually means how best to allocate money and people to achieve profitability goals. In the public sector, it may mean how best to allocate people and dollars to provide a desired level of service. The main idea here is that the resource is scarce and, as a manager, hard decisions have to be made about how to use it. More specifically, financial planning is setting budgets, making pricing decisions, and estimating future demand for products and services, in order to achieve profit and/or performance goals.

Why is formal planning necessary? First, of course, because a scarce resource (typically money) is involved. If we had enough money for everything, then we could simply raise our salaries and retire early. Secondly, it is very important to have general agreement within an organization about how goals are to be achieved. No assumptions should be made without clearly stating and documenting them. With a good financial plan, trouble signs can be spotted earlier and corrective action taken sooner. Businesses which fail to plan effectively are the best illustration of the need for planning.

Let me offer one last reason why planning is important. For many companies, planning is a necessity because of the complexity of their operations. A typical manufacturing company may purchase thousands of parts for use in a vast array of products, and assemble them in many different locations. They cannot wait until there is no money in the till to decide that it's time to raise prices. And the current rates of inflation make this an even more important consideration.

THE TYPICAL PLANNING PROCESS

Okay, let's assume that one accepts the need for financial planning. So what's the big deal? Well let's look at the typical planning process and

I'll show you.

First, planning involves lots of numbers. And these numbers change often. Financial planning involves projections into the future and is a very uncertain process. When you're uncertain, then you have to do contingency planning. Play "what if" games. What if sales are 20% higher than planned? What if the cost estimates are too optimistic? What if our product sales mix is different? Because of uncertainty, alternative plans are necessary, increasing the amount of work required to plan several times over.

And that's not all. The attempt to reach a targeted objective such as profit adds to the work. It may take several passes before all of the budgets combined with the sales estimates, cost estimates, and so forth, sum up to the desired results. The task soon becomes monumental.

The following is not an uncommon occurrence: You work many hours preparing budgets and doing sales forecasts. With a board meeting just a few days away, you finish your plan. The company president takes one look at the results of the combined numbers and gives it back, requesting a 15% cut in the budget. You prepare a revised budget, repeat all of the calculations, this time under increasing pressure to get the job done fast. The day before the board meeting, marketing revises the forecast. All of the budgets must be revised again. And now it is getting late into the evening the day before the meeting. The planning process finally ends. With a good plan? No, with exhaustion. Does this seem like a doomsday tale? It's not. I've seen this happen many times. No wonder people dread budgeting time.

Combine the sheer effort required to plan effectively with the requirements for a good plan: It must be TIMELY. In a dynamic, growing company, a plan must reflect today's expectations, not yesterday's. It must be ERROR FREE. Late-night, reworked plans suffer from simple calculation errors. Errors due to using the wrong set of estimates, because they keep on changing. Imagine the embarrassment of a summation error. And with all this, the plan must remain FLEXIBLE. I worked on a profit plan for a company a few years ago which added an entire product line between iterations of the plan. And finally, when you are all done, a good plan must be WELL DOCUMENTED. What factors were used for overhead? What was the basis for the final sales figure? How was a particular number calculated? All too often, there is little documentation on how a plan was actually prepared.

To summarize: A typical financial plan involves lots of numbers, which change often. The need for many iterations makes this process time consuming and exhausting. At the same time, the plan must be timely, error free, and well documented. In short, good financial planning is not easy.

WHAT IS THE BEST WAY TO PLAN?

Given that this is the nature of planning, what is the best way to plan? How can it be done with a minimum of difficulty? Traditionally, there have been two ways of planning. Planning by hand (and calculator) and planning using the computer. Let's take a look at both of these methods and evaluate the pluses and minuses of each.

Preparation of plans manually has several drawbacks. First, because of the amount of data involved and the number of iterations, it is slow and time consuming. After many iterations, accuracy becomes a problem. The wrong estimates may be used, particularly if they keep changing. Calculation errors seem to increase with each iteration. And documentation is usually not very good.

On the other hand we have financial planning on the computer using the traditional programming languages like BASIC, FORTRAN, or COBOL. Once set up, a model written in one of these languages will run on the computer in a matter of minutes or seconds. Great! But here's the catch. The model will run very quickly once it has been set up, but it may take months to get it developed. And you need a programmer. Let's see what can happen. You start your plan well in advance of the next budgeting cycle. With six months lead time you give a precise set of specifications to an enthusiastic programmer who dutifully sets about coding your model. At the end of the first three months, he comes back to you with his first try. You patiently point out where the model is not consistent with the specifications, settle on a set of revisions, and the model is reprogrammed to your satisfaction. All set, right? No. As you begin using the model, the company president starts to change his mind (even though he reviewed the original specifications). Add a decimal place here, another line item there. Why aren't all twelve columns of data on the first page? Frustration.

What is the moral of our story? Programming a planning application with the traditional programming languages lacks flexibility. The programmer needs lead time to set up the application and has difficulty in reacting to short term changes. How about adding another division to a multi-divisional company? Try changing every format statement in the model in an hour. And add to that the bother of documentation.

To summarize, manually prepared plans can be flexible, but they take a long time to do and lots of effort, especially if several passes are done. They often lack documentation. Planning with traditional programming languages takes too long to set up, is inflexible, and requires the services of a programmer.

PROBLEM ORIENTED LANGUAGES

Let me digress for a moment. For several decades now, computer scientists have been searching for a "universal" programming language. ALGOL? PL/I? APL? PASCAL? The search goes on. Each has its merits, each its disadvantages. But these "procedure oriented" languages have one thing in common: You have to be a programmer to use them. And it is altogether too easy to include bugs in even the simplest of programs. As long as there is a programmer acting as middleman between the user (or analyst) and the computer there are going to be communication problems. Maintenance problems. Resource and priority problems.

What's the answer? A planning oriented application language which incorporates the good aspects of traditional programming, but eliminates the problems. Where plans can be set up and revised easily, without having to be a programmer. What I am describing here is one example of another class of programming languages, "problem oriented" languages. Languages which have been designed to provide solutions in a general way to classes of problems. Simple enough to be used by non-programmers. Easier to debug. Self-documenting. QUERY is an example of a problem oriented language. It provides access to IMAGE data bases in a fashion simple enough to be used by non-programmers. Dollar-Flow is a problem oriented language, designed as a tool for non-programmers who want to set up tabular planning reports.

Financial planning is an area well suited to problem oriented languages. There is a considerable amount of generality in what planners do, although no two plans are the same. A financial plan typically involves mathematical operations on rows and columns of numbers. With well defined rules for the calculations. And the burden of planning in any other way gives the financial planner considerable incentive to try new appproaches.

This is a good start. But we still have to get the planner onto the terminal and communicating with the computer. How is this done? By giving him an effective tool. One which is both friendly and enables him to get the job done in a way that he understands.

DESIGNING FRIENDLY SYSTEMS

This leads us to the next point: What makes a system "friendly"? How can a system be designed so the novice or non-computer type feels comfortable with it? I ofer here a few of my ideas and techniques for developing friendly systems.

SIMPLICITY

Keep the system simple at all cost. Do not let the internal structure on the computer dictate how a system looks to the user. Let him express his ideas in his own terms. For example, the original design for the Dollar-Flow language was based on a set of documentation which I prepared for a group of accounting types. This documentation described the workings of a particular customized model on a line by line basis. I figured: What could be a better set of design specifications for a language than actual documentation? As you document your model you are also writing your program! Another example. Dollar-Flow re-orders calculation rules automatically. Thus, line 1 on a report can reference data on line 10, which, in turn, can reference data on line 20. Dollar-Flow automatically figures out the proper sequence for calculations (calculate 20, then 10, then 1) without any intervention by the user.

It is important that the application be self documenting. For example, Dollar-Flow is a menu driven system. At each step of operation, the user knows his alternatives. There is little need for a "pocket guide" to the language. This is not to say that there is no need for manuals. A good manual is important. But it is a fact that few people actually read manuals. The less a system forces a user to read the manual, the more usable it will be.

Not only should the user be told what his alternatives are, the system should also help him to choose the proper response. Throughout the Dollar-Flow prompts, the most likely response is shown in brackets as the "default" response. In some cases, he can use the default response without bothering to even understand the question! For example, the prompt:

USE STANDARD OVERALL REPORT FORMAT (<Y>,N,W-WIDE PAGE)?

In one brief prompt, the user can see his options and pick one. A simple carriage return will cause the system to use the default response. And his entire report format is set up. No PRING USING or FORMAT statements. Very simple. And it can be changed easily. As the user becomes more familiar with the language, he can begin to exercise more options. With an 'N' response, Dollar-Flow leads the user through a review of the many formatting alternatives. Report formatting can even be done on a trial and error basis. Start off with the standard format, then change the column width or number of decimal places shown as needs require.

As i already mentioned, the design for the Dollar-Flow calculation rules was based on a set of user oriented documentation. Ask a user to describe how the values on the report are to be calculated in his own terms. With the addition of a few uote marks here and there, he has already written a program in the Dollar-Flow language. Self-documenting languages not only save the effort required for documentation, but make debugging easier as well.

One last comment about simplicity. Save the user concerns about internal structure through structure independent (or data base) approaches to data relationships. One of the beauties of QUERY is that the user doesn't have to concern himself with all of the details of the data base to get a simple report. In Dollar-Flow, all reports are programs, all saved programs are files, and all save files contain reports. To reference data on a saved Dollar-Flow report, simply indicate the line name and the report save file name:

```
MARKETING BUDGET = 'BUDGET' OF 'MKTG';
```

There is no need for the user to know how the data is stored or even which line on the 'MKTG' report is the 'BUDGET' line which he is using.

ERROR HANDLING

Okay, so let's say you have implemented a simple system. Does this mean that users won't make mistakes? Of course not. In fact, the friendlier a system is, the greater the likelihood that the users will not be computer types. So, keep in mind that "too err is human, to forgive is good systems design." Of course, you must edit all inputs. But then use a friendly approach when the user has made an error. Because Dollar-Flow is menu driven, simple typing errors cause the system to repeat the prompt. Errors of a more complex nature, such as where a report is referenced but does not exist, generate intelligible error messages. Along with each error message give a message number. And provide a glossary with the documentation which gives even greater detail on the possible cause of the problem.

At the same time that it is informative, a system should help the user to work around problems. For example, in the case of an invalid report reference in Dollar-Flow, the user can interactively specify a different report name, or values, or zeroes. He can also indicate that computation should cease after a scan for further errors. Again, unless a particular error is extremely serious, warn the user and proceed (with his permission). Another example. As far as the mathematician is concerned, division by zero gives unworkable results. In Dollar-Flow, division by zero yields 'invalid' numbers (which print as asterisks), but doesn't stop computation. It's amazing how much more satisfying a user finds a report filled with asterisks than just a list of error messages. At least he can look at the format to see if it's to his liking.

If you must tell the user that he has made an error, tell him as early as possible. One of the most enlightened things done by the MPE operating system is to edit the job statement when a job is being streamed from an interactive session. It sure is better to find out right away than waiting for the job to begin execution to find out that a simple error has been made on the JOB statement. Report development in Dollar-Flow is completely interactive. If a user is setting up a report and he enters a calculation rule with invalid syntax, the system responds with a message immediately, and permits him to edit his error (not unlike the BASIC interpreter). It is not necessary to go into the computation step to find many errors.

MAINTENANCE AND SUPPORT

Let us assume that as an enlightened designer of friendly systems you have now designed and implemented your masterpiece. Are you done? Of course not. This is only the first step. There are two more important aspects which are critical for good, friendly systems: Continuing improvement and good support. Let me talk about continuing development first. No system is great on the first try. I am a believer in the iterative approach to systems development, if you can afford it. I am not talking about sloppy design. I am talking about the tremendous wealth of ideas that you can get from your users, AFTER you have implemented a system. Try to be receptive to the suggestions of your users (even if they are infeasible). Never give a critical user the impression that you think he has just offered a bad idea. Go out of your way to solicit ideas from your users. If the situation merits it, get involved in several of their applications. You can learn about ways the system is being used that you never thought about. Ways in which its use may be awkward. Which messages are more annoying than useful. Which features are badly needed. I send periodic questionnaires to my users (some of them even respond). This helps to prioritize new features. And users group meetings are a great boon to information flow.

How should this wealth of new ideas be integrated into an already developed system? Carefully. Do not rush a new version of a system out to users just because they need a particular feature. You must let a new version of a system be "burned in" first by a test site. Software bugs cost you credibility. Once lost, credibility is very difficult to reestablish, so reliability is extremely important. After all, would a user prefer a system with the bells and whistles he wants but doesn't work, or one which works with a few less features?

Speaking of bugs and user suggestions leads me to the question of support. There is nothing more frustrating to a user than to get 95% of the way to his computer solution only to be stopped by the application package he is using. For any reason. If you can afford to do it, good support pays great dividends. Dollar-Flow is supported in an "on-line" fashion. This means that if a user has a problem, he picks up the telephone and calls. If his problem is with an existing report, we may even log onto his system and take a look at that report. This kind of support not only helps to find and eliminate system problems quickly, but we also find out about areas where the documentation may be confusing (or incorrect). Where another feature might simplify the user's application. In short, on-line support can be another source of good ideas from users.

Let me summarize these techniques for creating friendly systems. First KEEP IT SIMPLE. Try to think like the user instead of a computer expert. Use his terms. Assume that he won't read the manual. Try to make it self-explanatory. Second, be INFORMATIVE but FORGIVING with your error handling. Edit all inputs, but don't bother the user with minor errors. When the application merits, CONTINUING ENHANCEMENT will make a much more usable system. Respond to user suggestions. But exercise good judgment in the trade-off between adding new features and degrading SYSTEM RELIABILITY.

PROFIT PLANNING

I am not going to take too much time on the last part of my talk. I am just going to show you a few sample reports prepared using Dollar-Flow. At the risk of violating my agreement not to make a sales pitch, I invite you to visit the PALO ALTO GROUP's booth during the vendor exhibits for a demonstration of Dollar-Flow in action.

Let me first describe the typical company profit planning cycle and the environment in which a planning tool like Dollar-Flow is used. The typical Dollar-Flow user is the accountant or company controller who is responsible for preparing the reports. Not a programmer. Most users are working on in-house HP3000 systems. With access to CRT's and a system line printer nearby. Reports are written interactively, and manual inputs are also entered via the terminal. Usually, reports are printed on the CRT for review then saved when the user is satisfied with the report. If hard copy is desired, the reports can be routed to the line printer. For generating large numbers of reports, the "batch command mode" is used, where with very little terminal input a large number of reports can be generated.

Profit planning typically begins with a preliminary sales forecast. Preliminary. Sales forecasts always change. And at the last minute, too. Often the sales forecast is done on a product-by-product basis for the first year or so, then combined with overall dollar sales projections further in the future. The near term unit forecasts are sometimes adjusted based on an overall dollar figure. The forecast is iterated several times. To make a change, the product manager just runs Dollar-Flow, inputs whichever figures have changed, pushes a few buttons, and the new sales forecast is ready. Since many parts of the profit plan depend on this sales forecast, the typical plan is usually set up with reports referencing the sales forecast report. If the figures are changed on the sales forecast, these changes will be automatically reflected on the other reports the next time they are run. Some manufacturing companies even use a multi-level sales forecast step, where a build plan (or production plan) is generated from the sales forecast.

Meanwhile, departmental budgets are prepared. Some Dollar-Flow users centralize the budgeting function and only distribute budget worksheets to each department or location. This is usually done if there are only one or two budget iterations. On the other hand, some of our customers distribute the budget preparation, with each location setting up its own budget in Dollar-Flow. In this case, figures can be input to Dollar-Flow, changes can be made, and several iterations of the budget can be done all in a matter of minutes. And budget consolidations are fun! With a few simple commands to Dollar-Flow, a whole series of budgets can be consolidated into a departmental or divisional budget. When changes are made to the low level budgets, they automatically are reflected on the consolidated budget the next time it's run.

The profit/loss projection is next. Using the data from the sales forecast, the build plan, and the budgets, and adding factors for items like sales discounts and returns, a pro forma operating statement is prepared. Often, the bottom line (profit) on this report determines what (if any) changes need to be made to the budgets. With a flexible tool like Dollar-Flow, a financial executive can even do sensitivity analysis: What if sales are 20% lower then forecast? What if our discount schedule is more aggressive and our volume is larger?

Some companies that rely on substantial amounts of debt to finance their operations combine the profit/loss projection with a cash flow projection. This is because interest paid (an item of expense on the profit/loss statement) has an impact on the amount of money required to run the business. This determines the level of borrowing, which, in turn, affects the amount of interest which is paid. Dollar-Flow, and most good financial planning languages, can solve the "simultaneous equations" this circular logic represents, and determine a level of debt and debt service which are consistent with each other. This is far more difficult when done manually.

Another procedure which is laborious when done by hand is the aging of accounts receivable and accounts payable projections. Using Dollar-Flow, once the rules for aging have been set up, a change in the sales forecast or the build plan will automatically be reflected in new receipts and payables projections.

And, finally, some companies prepare pro forma balance sheets as the last step in their profit planning cycle. This is not necessarily the way all companies plan. Or even the way all Dollar-Flow users plan. In fact, many Dollar-Flow users are not even responsible for profit planning. Instead, the system is used for a wide variety of ad hoc applications involving calculations on rows and columns of numbers. It is even used as a design tool for systems which will later be hard-coded in COBOL, FORTRAN, or BASIC.

Some of the other applications of Dollar-Flow that I am aware of include:

Product pricing. Comparing alternative prices for a single product (the plotting capability is great for comparisons). Or comparing profit percentage across an entire product line. Financial ratio analysis. Comparing selected financial ratios against industry standards or company objectives. Capital budgeting. Rates of return and discounted cash flows can be calculated easily using built-in financial functions.

Performance reporting. Variance reports showing actual budgets or profits versus plan. How sales are doing against target. (One Dollar-Flow user generates 500 graphs every month showing product line sales performance for every branch of every distributor who markets his products!)

SUMMARY

Let me leave you with a few parting thoughts. Financial planning is not an easy process. Figures change. The whole approach to a plan may change. And you need your results yesterday. Traditional systems design and programming methods are not going to be effective in this kind of situation. Use a better approach. With a friendly, problem oriented planning language like Dollar-Flow, applications nightmares can become applications successes.

A DIRECT MAIL ADDRESSES APPLICATION IN A TRANSACTION PROCESSING ENVIRONMENT WITH SHORT RESPONSE TIME REQUIREMENTS

TOIVO TAGEL
DAFA

0 TABLE OF CONTENTS

1 Introduction and background
2 Problem description
3 Alternative solutions
4 Conceptual system design
5 Follow-up of user satisfaction

1 INTRODUCTION AND BACKGROUND

The Swedish postal authority maintains an independent business division, POSTENS ADRESSREGISTER (PAR), involved in providing addresses of Swedish companies and organizations for direct mail purposes. For this division a direct mail addresses data base complete with application programs was developed on a HP 3000 Series III.

The addresses of the data base can be selected and retrieved in a real time dialogue using logically complex search criteria for qualifying the desired output. The selected addresses can then be printed. The system has been designed to allow a daily output of 100 000 addresses during office hours.

The data base has capacity for 600 000 addresses of firms and organizations. In addition there is space for 200 000 names of key management staff, e.g. managing directors, financial managers, of those firms. The data base has been particularly designed for this application using a combination of several KSAM and MPE files.

Each address in the data base is described by a number of search items such as telephone area code, postal number (ZIP), number of employees, annual business turn over as well as an arbitrary number of selection codes describing the line of business of each company,

Addresses may be retrieved according to any logical combination of the above mentioned search items. Thus it is possible to retrieve e.g. all financial managers of firms within a given annual turnover range, in a certain geographical area and belonging to a specified series of business lines.

The system is characterized by very efficient avaluation of complicated retrieval criteria. This provides short response times which is important for the users of the system. It is e.g. possible for a salesman to have a telephone conversation with a customer and at the same time using the system for inquiries.

Maintenance of the data base is carried out interactively from user terminals. Individual addresses can be retrieved at any time, amended and placed in an update file for batch updating at the end of the day.

The system has been designed and developed by DAFA in collaboration with HP Sweden. DAFA is the Data Center for Administrative Data Processing, a Swedish government authority providing data processing and consulting services within the state administration.

2 PROBLEM DESCRIPTION

2.1 Business requirements
2.2 Previous ADP systems
2.3 Statement of objectives
2.4 Definition of requirements

2.1 Business requirements

There are some important factors that determine the competitivness and success of companies in the direct mail business. The ones that are considered most important are:

Accuracy of addresses
Completeness of data base
Correctness of address count
Service level of operations

The accuracy of the addresses is perhaps the most important individual factor. Every time a company changes its mailing address, a corresponding change must be made in the addresses data base. The purpose is, of course, to keep the number of undelivered mail and by reason of that wasted adveritizing material at a minimum. As there are close to 400 000 firms in Sweden this problem requires a lot of follow-up work taking into consideration that new companies are established daily and old companies disappear at the same rate.

Also of the same importance is the completeness of the data base. It is a basic requirement of customers of direct mail firms that the addresses they order and pay for cover all existing firms and organizations. There is today in Sweden no practical way of collecting addresses automatically. The work has to be done manually from whatever sources that can be found. In this tedious task it must be possible to check the existence of an address in the data base. If it is there the correctness is checked; if it is not there the new address is added to the data base.

In the direct mail business it is common practice among customers to ask for the estimated number of addresses of alternative target groups and target group combinations. This is usually done before an order is placed. The purpose is twofold. The customer wants to print the same number of advertizing material as there are addresses in his particular address selection. And he wants to buy the addresses from that direct mail firm who has the most complete coverage of the selected target groups. It is thus necessary to be able to retrieve the current count of addresses for any combination of selection criteria.

The mentioned problems are often discussed between the customer and the addresses firm salesman. They work out suitable target group combinations and plan the details of the customer's direct mail drive. In this process the salesman needs facilities for quick retrieval of current count information.

There are also other requirements related to the level of service that the direct mail firm wants to provide.

It must be possible to print addresses in different formats and on different stationary, adhesive lables etc. Sometimes the addresses must be delivered on the same or the following day. This means that a computer based system must as far as output be very flexible as to format, sorting order and posess a satisfactory retrieval and printing capacity.

2.2 Previous ADP systems

Pad had for several years been using a batch oriented ADP system which was run om a large IBM computer at the Post Office data center. The old system had specifically been developed for PAR. Unfortunately the system never was very successfull. The main shortcomings were:

- Long turn-around times for addresses count information making this facility practically impossible to use.

- Monthly updating of the data base resulting in slow and difficult maintenance work.

- Duplicate and inaccurate addresses as there were no simple ways for scanning the address file for such errors.

- The generally low service level which is inherent to batch systems which are meant to support work the nature of which is transaction oriented.

So, very soon after the initial contacts were taken in 1977 PAR agreed to collaborate in a study of possible improvements to their ADP operations.

2.3 Statement of objectives

In the beginning of 1979 the objectives and basic requirements of a new system had been worked out.

The fundamental objective was that the new system had to support the users, i.e. salesmen, maintenance staff and production people, in the bast possible way. The system had to be particularly designed to the processes the users were involved in.

This philosophy led to the conclusion that the new system had to be a terminal oriented transaction processing system. The following objectives were to be realized:

- Improved accuracy of the addresses through continuous revision and daily updating of the data base.

- Improved efficiency of the revision work with terminals for reviewing the data base content and for entering emendments and new addresses.

- Real time facility for retirieving the number of addresses in a given target group defined by any logical combination of retrieval criteria.

- Simplyfying retrieval and printing of addresses from terminals used for entering output requests and formatting information.

- Future capacity for adding additional applications.

2.4 Definition of requirements

It was required that address retrieval be possible using any one or an arbitrary combination of the following search items:

Target group codes (0-27 per address)
Administrative district code
Telephone area code
Postal district
Region code
Annual turnover
Number of employees

In addition for the above keys the following search items were required for address revision purposes:

Reference number
    000 001-600 000 for address records
    600 001-800 000 for name records
Organization short name
Organization number

The volume and performance requirements were the following:

600 000	address records
200 000	staff name records
20 000	address amendments per month
30 - 40	printed retrievals per day
100 000	printed addresses per day
10	simultaneous terminal sessions
15	maximum number of terminals
1 000	terminal transaction during peak hour

3 sec	maximum response time for 90 % of terminal transactions
1 sec	maximum response time for 90 % of error messages

200 hrs	effective use time per month
4 hrs	maximum allowed down time
95 %	minimum availability per month

The above requirements were part of the user requirement specifications which were prepared for inclusion in the tender invitations.

3 ALTERNATIVE SOLUTIONS

3.1 Tender invitations
3.2 Mini-computer solution

3.1 Tender invitations

At this point in time it was not yet decided whether the system was to be run on a service bureau or on dedicated equipment installed at PAR. The decision was to be made when the final costs for the alternatives were known.

So, tender invitations were sent to the Post Office computer center, DAFA amd a number of computer manufacturers.

DAFA proposed a solution on their IBM 370/158 and a software packaga STAIRS also from IBM. The intention was to supplement STAIRS with tailor-made software for updating and **for** editing and printing the output. Terminals were to be installed at PAR and connected with DAFA over leased telephone lines.

Although the proposal was technically sound, the quoted price was not very competitive. Development cost was 285 000 Swedish Kronor and the annual production cost was estimated to 2 400 000 Kronor.

3.2 Mini-computer solution

DAFA, although participating in the competition with its own service bureau proposal, was asked to assist PAR in evaluating the tenders from the mini-computer manufacturers.

Finally HEWLETT PACKARD was chosen for delivering the necessary computer system and DAFA for providing all application development and project management. The following computer hardware was ordered:

HP 3000 III 512 kbytes
4 disks 120 Mbyte
1 disk 50 Mbyte
2 tape drives
1 console 2635
1 printer 600 lpm
10 terminals 2640

It was decided to use COBOL and SPL for application programming and KSAM for data base management.

Investment in Swedish Kronor:

1 620 000	Computer system, terminals
680 000	Application development
2 300 000	System investment
400 000	Computer room, offices
2 700 000	Total investment (approx.)

Annual cost in Swedish Kronor:

700 000	Investment cost (15 %, 5 years)
160 000	Computer system maintenance
480 000	Operating staff, application maintenance
1 340 000	Total annual cost (approx.)

It is interesting to note that the annual cost of approximately 1.3 million Kronor is very much less than the quoted price from DAFA asking 2.4 million kronor for the same service in its service bureau proposal.

4 CONCEPTUAL SYSTEMS DESIGN

4.1 System description (functions)
4.2 Key inputs and outputs
4.3 System flowchart
4.4 Data base design
4.5 Response time analysis
4.6 Batch throughput analysis
4.7 Dialogue language
4.8 Application logic

4.1 System description (functions)

The business requirements of PAR as they have been implemented in the system comprise functions for:

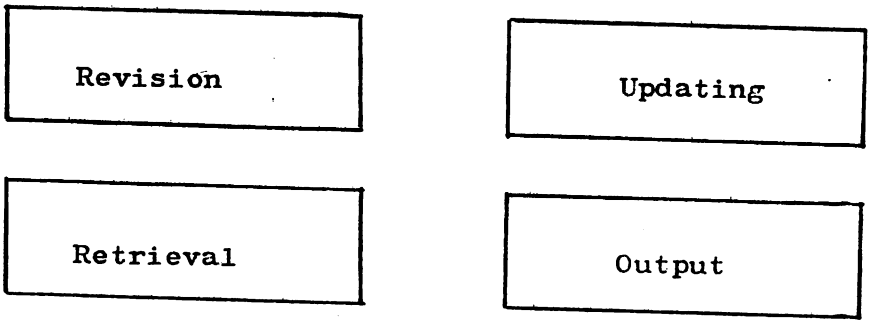
- Interactive revision of addresses including facilities for entering amendments and new addresses.

- Updating the data base in batch mode, i.e. adding amendments and new addresses to the data base from a transaction file.

- Selecting addresses that correspond to a given combination of search criteria and retrieving the number of addresses in the found set.

- Entering editing and ordering specifications and requesting printed output (or on magnetic tape for off line printing) on the selected address set.

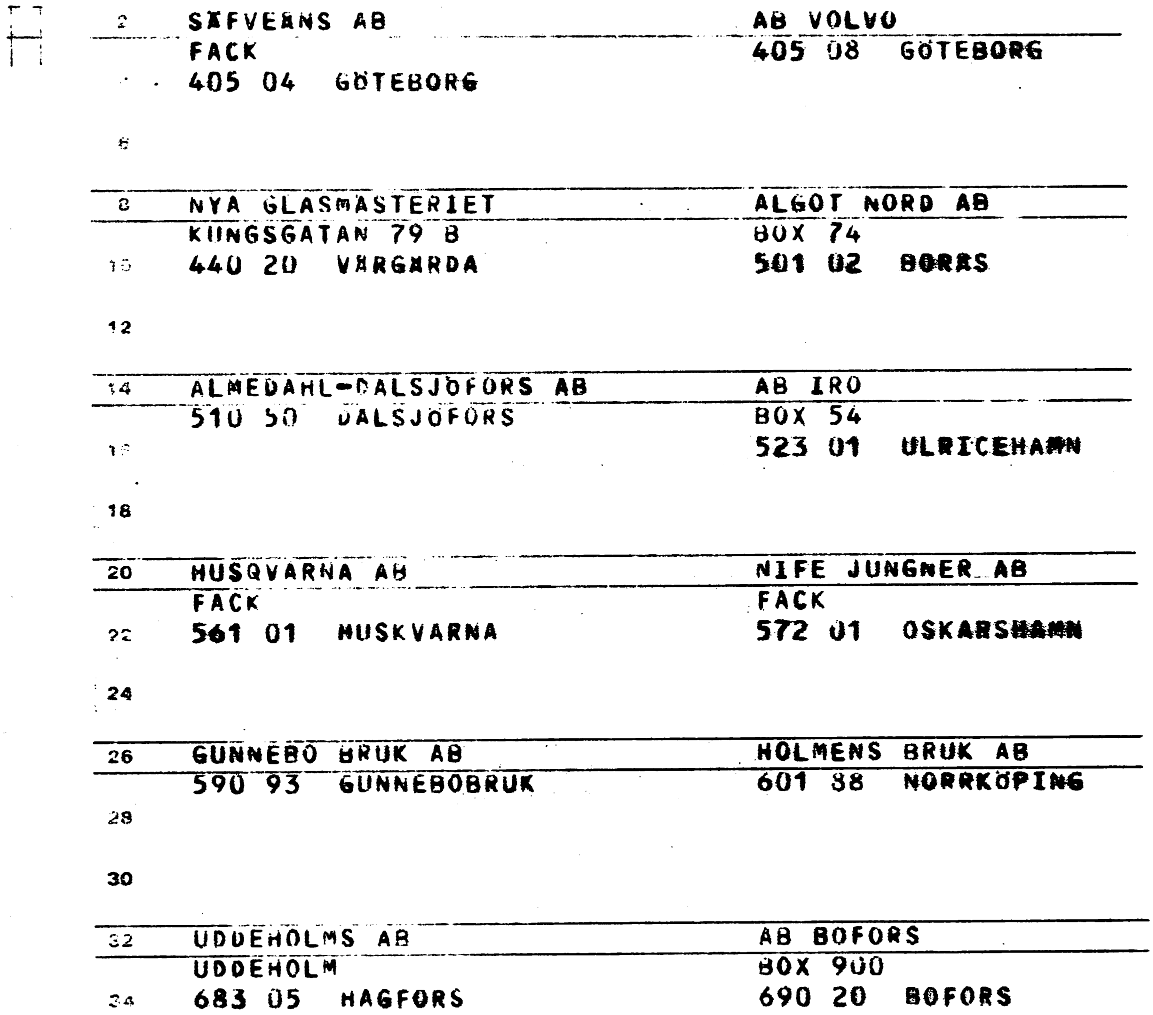
The above mentioned four functional requirements correspond to the following sub-systems of PARAD:



In addition there are functions for taking back-up copies of the data base and restoring the data base in case of damage.

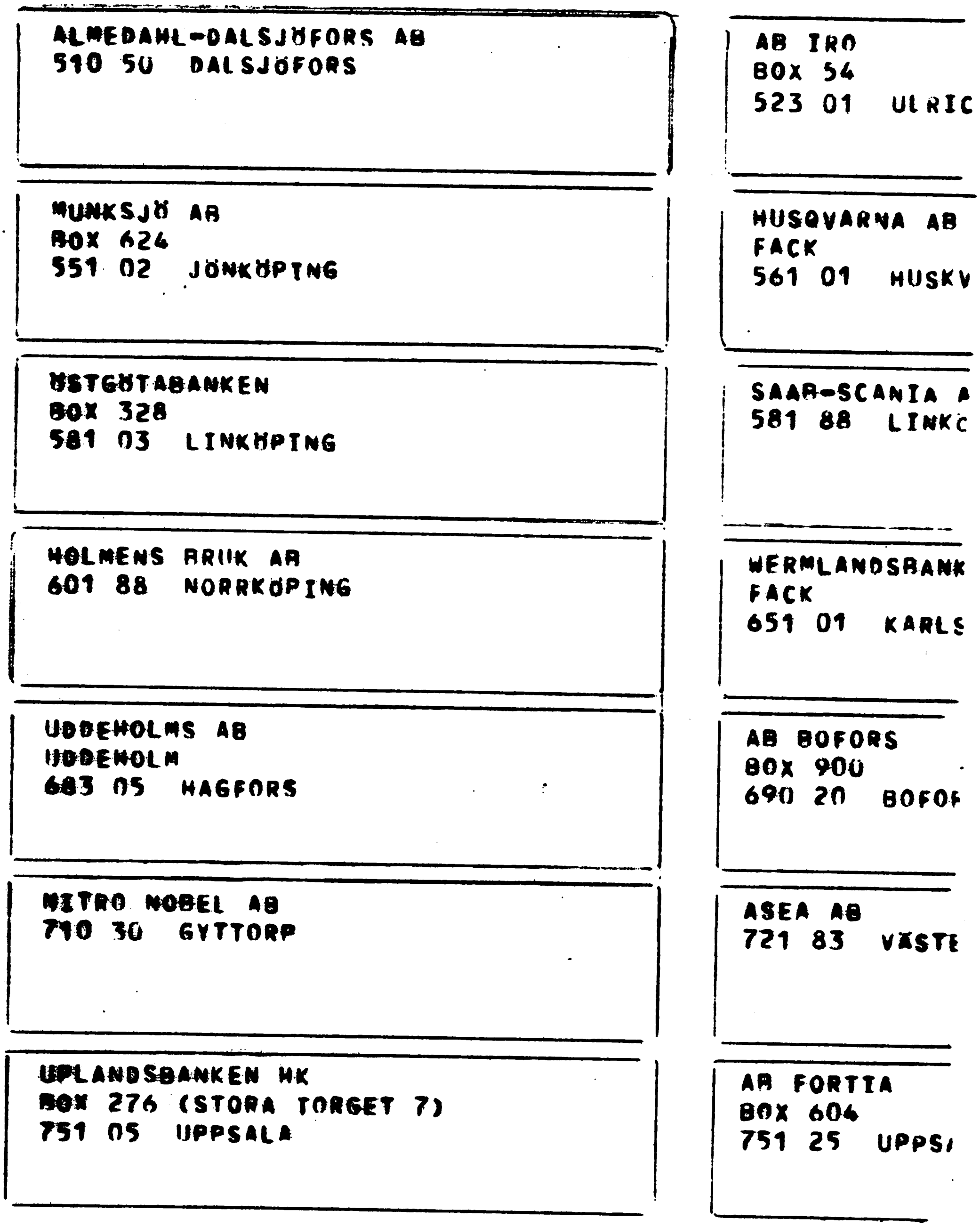
4.2 Key inputs and outputs

This and the following pages show typical output from the system:



```
SÄFVEÅNS AB                          AB VOLVO
FACK                                 405 08   GÖTEBORG
405 04   GÖTEBORG


NYA GLASMÄSTERIET                    ALGOT NORD AB
KUNGSGATAN 79 B                      BOX 74
440 20   VÅRGÅRDA                    501 02   BORÅS


ALMEDAHL-DALSJÖFORS AB               AB IRO
510 50   DALSJÖFORS                  BOX 54
                                     523 01   ULRICEHAMN


HUSQVARNA AB                         NIFE JUNGNER AB
FACK                                 FACK
561 01   HUSKVARNA                   572 01   OSKARSHAMN


GUNNEBO BRUK AB                      HOLMENS BRUK AB
590 93   GUNNEBOBRUK                 601 88   NORRKÖPING



UDDEHOLMS AB                         AB BOFORS
UDDEHOLM                             BOX 900
683 05   HAGFORS                     690 20   BOFORS
```

Addresses printed on normal computer output stationary



ALMEDAHL-DALSJÖFORS AB
510 50 DALSJÖFORS

MUNKSJÖ AB
BOX 624
551 02 JÖNKÖPING

ÖSTGÖTABANKEN
BOX 328
581 03 LINKÖPING

HOLMENS BRUK AB
601 88 NORRKÖPING

UDDEHOLMS AB
UDDEHOLM
683 05 HAGFORS

NITRO NOBEL AB
710 30 GYTTORP

UPLANDSBANKEN HK
BOX 276 (STORA TORGET 7)
751 05 UPPSALA

AB IRO
BOX 54
523 01 ULRIC

HUSQVARNA AB
FACK
561 01 HUSKV

SAAB-SCANIA A
581 88 LINK

WERMLANDSBANK
FACK
651 01 KARLS

AB BOFORS
BOX 900
690 20 BOFO

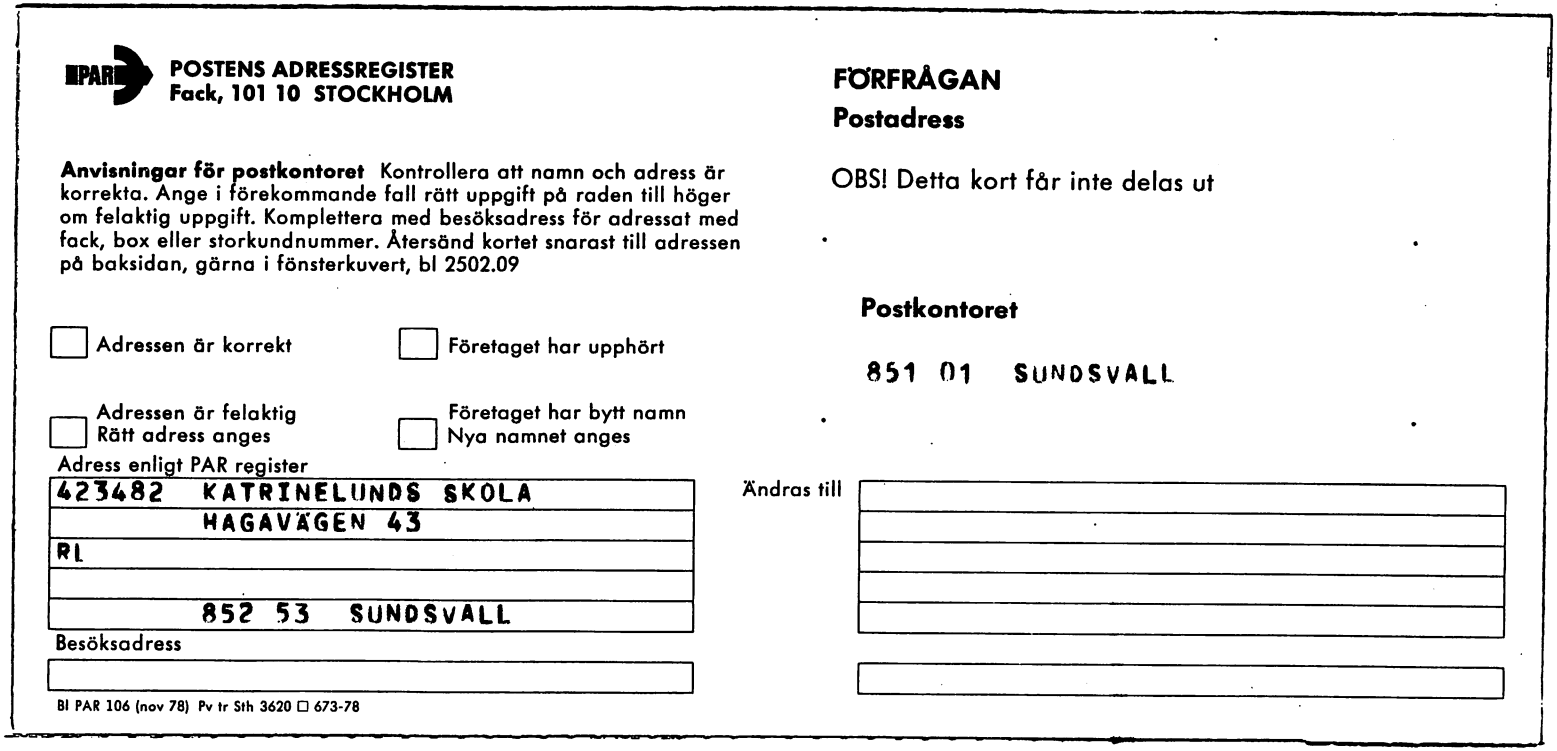
ASEA AB
721 83 VÄST

AB FORTIA
BOX 604
751 25 UPPS

Addresses printed on adhesive labels

P34-12

Inquiry card used in the revision work



POSTENS ADRESSREGISTER
Fack, 101 10 STOCKHOLM

FÖRFRÅGAN

Postadress

OBS! Detta kort får inte delas ut

Anvisningar för postkontoret Kontrollera att namn och adress är korrekta. Ange i förekommande fall rätt uppgift på raden till höger om felaktig uppgift. Komplettera med besöksadress för adressat med fack, box eller storkundnummer. Återsänd kortet snarast till adressen på baksidan, gärna i fönsterkuvert, bl 2502.09

Postkontoret

851 01 SUNDSVALL

☐ Adressen är korrekt

☐ Företaget har upphört

☐ Adressen är felaktig
Rätt adress anges

☐ Företaget har bytt namn
Nya namnet anges

Adress enligt PAR register

423482 KATRINELUNDS SKOLA
HAGAVÄGEN 43
RL

852 53 SUNDSVALL

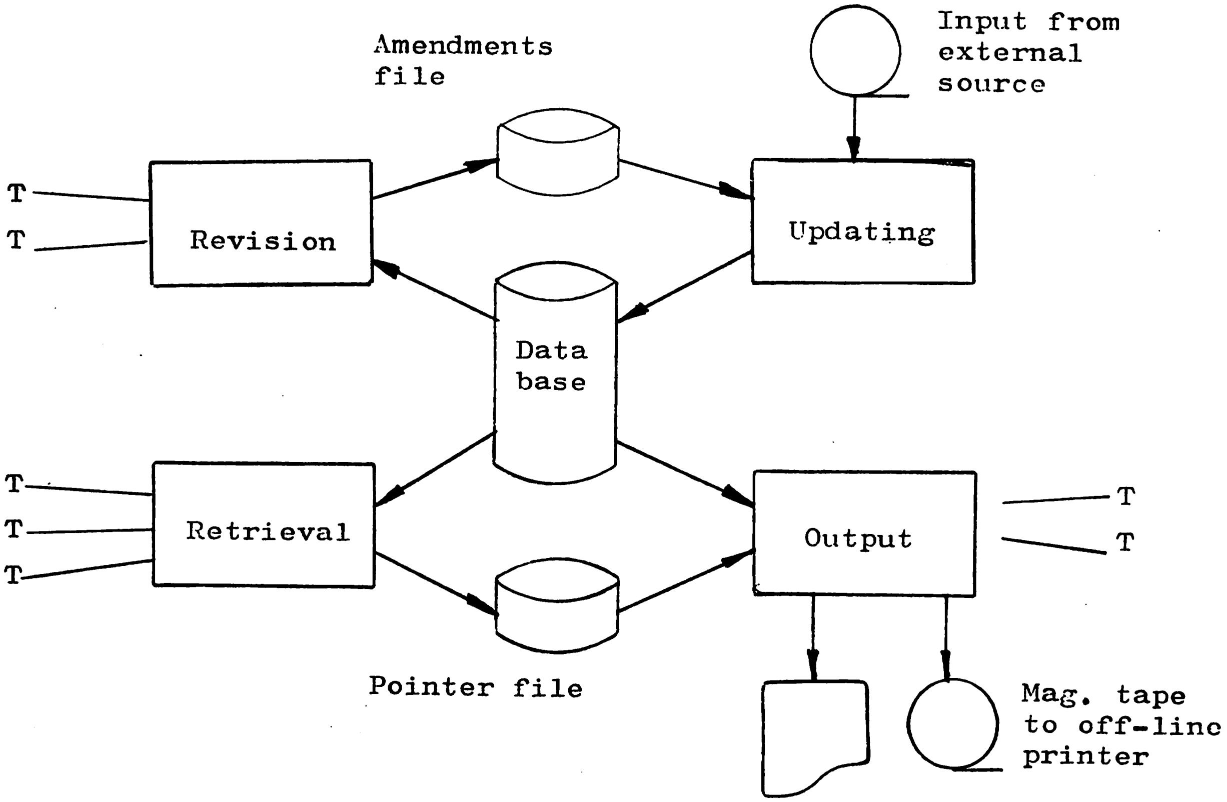
Ändras till

Besöksadress

Bl PAR 106 (nov 78) Pv tr Sth 3620 ☐ 673-78

4.3 System flowchart

The four sub-systems relate to the data base according to the following flow-chart:


Amendments file
Input from external source
T
T
Revision
Updating
Data base
T
T
T
Retrieval
Output
T
T
Pointer file
Mag. tape to off-line printer

The amendments file contains amendments and new addresses that are entered from the terminals involved in revision work.

The amendments file and possible input of addresses from external sources are processed daily in a data base update run.

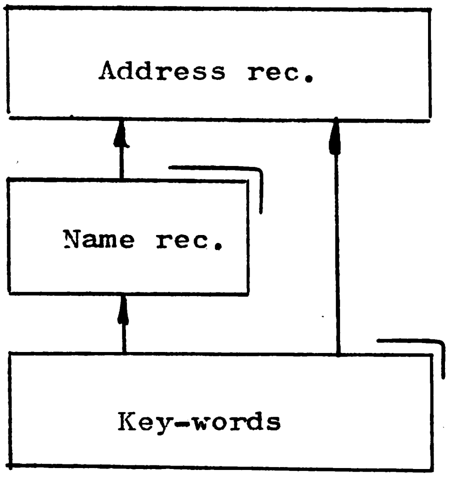
Tha address data base is a collection of several KSAM and MPE files and is described in the next section.

A pointer file is created as a result of evaluating address selection criteria. Each selection process creats its own pointer file.

The pointer files are input to the output sub-system and used there to indicate which particular addresses are to be retrieved from the data base and printed.

4.4 Data base design

The following diagram represents the logical structure of the addresses data base:



The address record holds the following data:

Reference number (master id.)
Organization number
Company name
Company address
Visiting address
Telephone number

The name record contains:

Staff member name

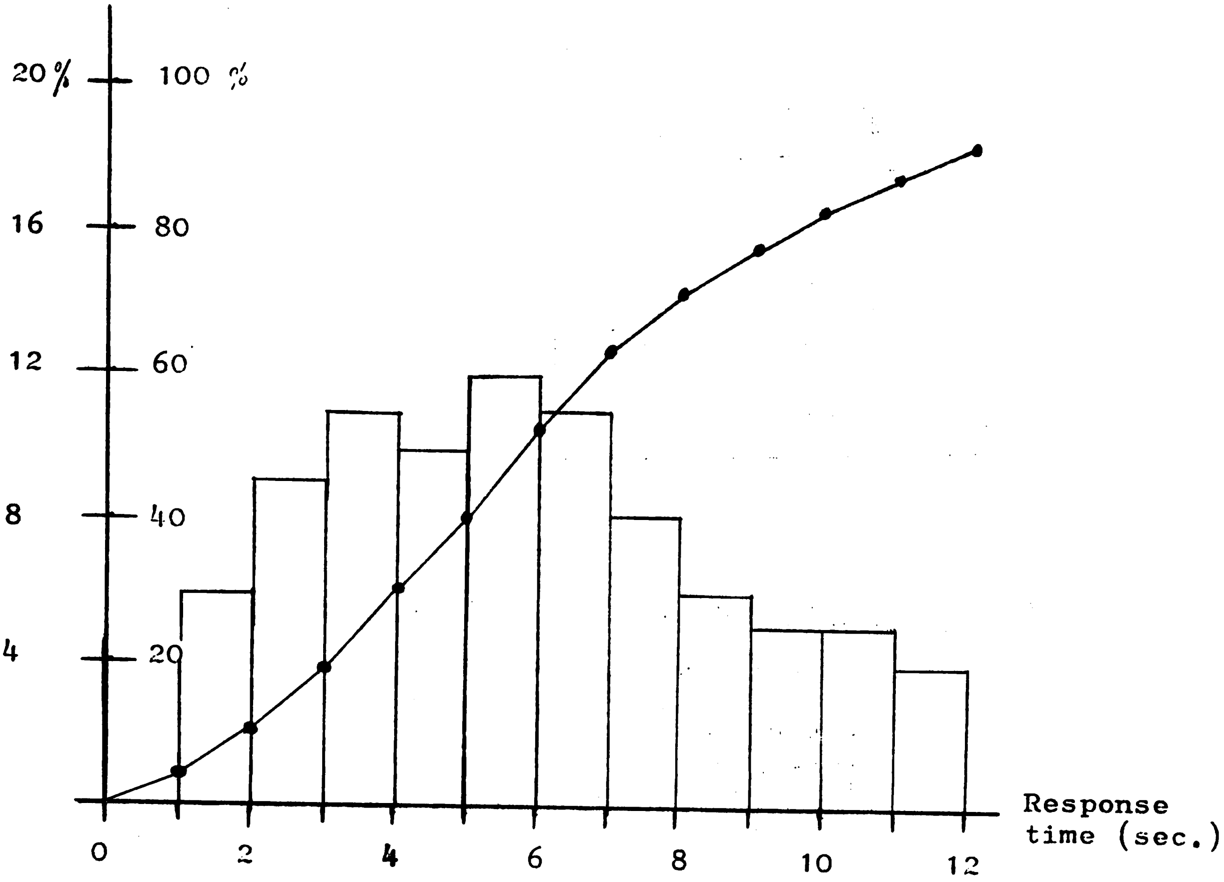
The key-word record contains:

Address target group codes (0-27 per address)
Staff mamber target group codes (0-3 per member)
Administrative district code
Telephobe area code
Postal district
Region code
Annual turnover code
Number of employees code

The physical implementation of the data base had to take response time requirements and batch throughput capacity into consideration. After a thorough analysis the physical data base design was established. (The data base design will be discussed in detail at the presentation of this paper. It is not included in the documentation.)

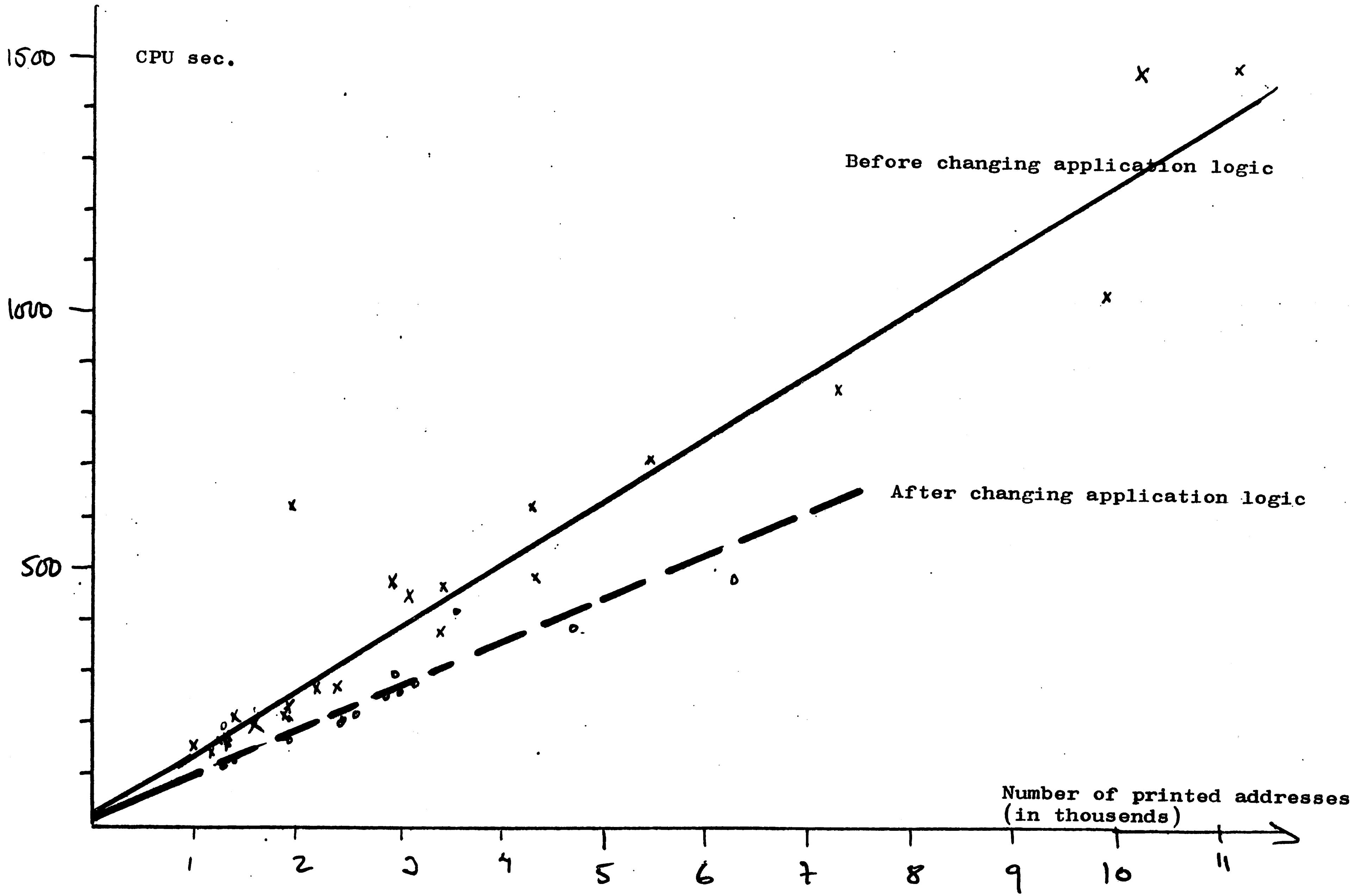
4.5 Response time analysis

The true response time for selecting addresses and retireving count information of the found set has been measured during normal work conditions taking 1 500 inquiries into the evaluation. The response times show the following distribution:



4.6 Batch throughput analysis

On the following page the output capacity has been measured and plotted as a function of the number of printed addresses:



1500
CPU sec.
1000
500
Before changing application logic
After changing application logic
Number of printed addresses
(in thousends)
1
2
3
4
5
6
7
8
9
10
11

4.7 Dialogue language

To facilitate user interaction with the PARAD system an interactive language was developed. The language comprises statements for

Selecting addresses from the data base
Specifying and producing the printed output
Revising and amending addresses

The following sample statements explain the nature of the language:

```
:SELECTION
COMPANY OR STAFF ADDRESSES
?C
?M.47110-47119,48200
?M.49000
?AND
?T.08,031,040
?END
999 999 REFERENCES FOUND
TYPE CUSTOMER ORDER NUMBER FOR OUTPUT
?JB4711

:OUTPUT
TYPE OUTPUT SPECIFICATION
?O.JB4711,L1=FINANCIAL MANAGER
?A.24
?END

:REVISION
?R.123456          Will display the full address and
                   related information of the company
                   referenced by the number 123456

?N.ESSELTE         Will display the company called
                   ESSELTE or companies whose name
                   start with those characters
```

In addition to the language commands functional keys are used for:

Browsing forward and backward
Switching between address and name displays
Ordering inquiry cards to be printed
Adding and deleting addresses
Etc.

4.8 Application logic

(This part is not documented but will be discussed at the presentation of this paper.)

5 FOLLOW-UP OF USER SATISFACTION

5.1 User satisfaction
5.2 Remaining problems
5.3 Future development

5.1 User satisfaction

The management and staff of PAR have at several occations expressed their great satisfaction with their new system. They feel they have got the necessary tools for reaching the initially stated objectives and believe that the new system is decisive for PAR's ability to compete on the direct mail market.

The project was completed in time and there were no cost overruns for the development and implementation work. As this is not often the case for ADP projects it certainly combined to the overall success of the project.

5.2 Remaining problems

There are very few remaining problems as most have been dealt with during the devolopment work. However, a number of good ideas have emerged during the work and PAR is now inte process of implementing improvements to futher make it easier for the users to interact with the system.

5.3 Future development

The management of PAR is already planning for new applications for their computer. Specifiacations have been prepared for order processing, invoicing and accouts receivables.

DATA COMMUNICATION

by Heinz Jurack

INTRODUCTION

The data communications world has begun to undergo a major change in recent years , from an ad hoc array of "home brew" systems that worked more in spite of, than due to, publicity available facilities, to a more planned and integrated set of facilities designed both for communications of data and for public use.

At the same time, the data communications needs of users have been growing, measured in terms of sophistication, volume and geographic coverage.
In an age of transcontinental and multinational business, users require access to a full range of services and coverage.
Thus, the same user demands that lead to the development of high capability public data networks will inevitably also lead to the interconnection of these data networks.

ENVIRONMENT

The current data communications networking environment is characterized by unprecedented innovation and growth. New technologies are moving from the laboratory to commercial application in record time, and a variety of new service offerings are being presented to customers. The result is both widened opportunities for users to avail themselves of these new services and confusion on the part of carriers and customers alike as to the service effectiveness and economic viability of all the new offerings.

THE PROBLEM

The basic problem is to device a strategy for the interconnection of computer communications networks that is workable and acceptable for users, carriers and regulatory authorities.

- USERS seek solutions that are technically efficient, easy to use, and which enable them to take full advantages of all facilities on any of the networks.

- CARRIERS also seek solutions that are efficient, though they are unwilling to alter the internal operation of their systems. Any system for interconnection must also permit them to protect the integrity of their own network and to account to all services supplied .

- REGULATORY AUTHORITIES seek methods that are understandable, controllable and for which tariffs can be devised. Not all of these goals are congurent and compromises will need to be made along the way.

TECHNOLOGICAL BACKGROUND

A clear understanding of basic networking concepts is necessary to appreciate the issues in network interconnection.

- INTERFACE COMPONENTS

  An interface, simple enough,is something between two systems, devices or components that serves to connect them.
  The interface may be a system, device or component itself, or a set of specifications to which the connected things adhere.

  Computer communications interfaces are generally composed of both devices or components and specifications.
  There are many different aspects or ways of describing a computer communications interface.
  One possible breakdown, and the one chosen here, is to destinguish between different levels of function, viz., physical, electrical, logical procedural.

- PHYSICAL

The physical portion of the interface specifies the way in which the two devices are actually connected mechanically. This includes the number of wires and the dimensions of the physical connectors in which the wires terminate.

- ELECTRICAL

The electrical portion of the interface specifies the voltage levels and duration (or for some interface specifications the current flow) to be used for signaling on the various leads.
The basic capability provided by adherence to the standards at this level is the transfer of data bits across the interface. These bits may be identifiable as parts of characters and/or be used for higher level signaling functions within the interface.

- LOGICAL

The logical portion of the interface specifies how the data bits and/or characters are grouped into fields for the purposes of signaling and data transfer.
In a sense, the logical specification of a computer-communications interface provides a language that may be employed for controlling and effecting data exchange across the interface.
STANDARDS at this level are sometimes called "elements of procedure".

- PROCEDURAL

If the logical level of the interface is viewed as specifying the syntax of the data flow across the interface, then the procedural specifications should be viewed as providing the semantics.
Specifications at this level determine the legal sequences of communications control characters, or the legal contents of various fields, or the valid commands and responses in controlling data flow.

The same basic set of control characters or fields may be used in a variety of different ways according to the procedural specifications.
Standards for these different ways are sometimes called "CLASSES OF PROCEDURE".

PROTOCOL

The term "protocol" is generally used to refer to the logical and procedural aspects of an interface.
Thus, a protocol specification includes both syntax and semantics.
Semantics also include relative timing information - not just the legal commands, but who can issue them and when.
A complex interface may contain several levels of protocol. An appreciation of this aspect of protocols is quite important in designing and evaluating network interconnections.

It has been suggested that the term "protocol" be used to designate communications conventions between entities at the same level while "interface" be used to designate conventions between entities at adjecent levels.
This is an attractive suggestion, and some confusion may be eliminated if the use of these terms in this way becomes widespread.

HP'S DATA COMMUNICATIONS PRODUCTS OVERVIEW

HJ 1

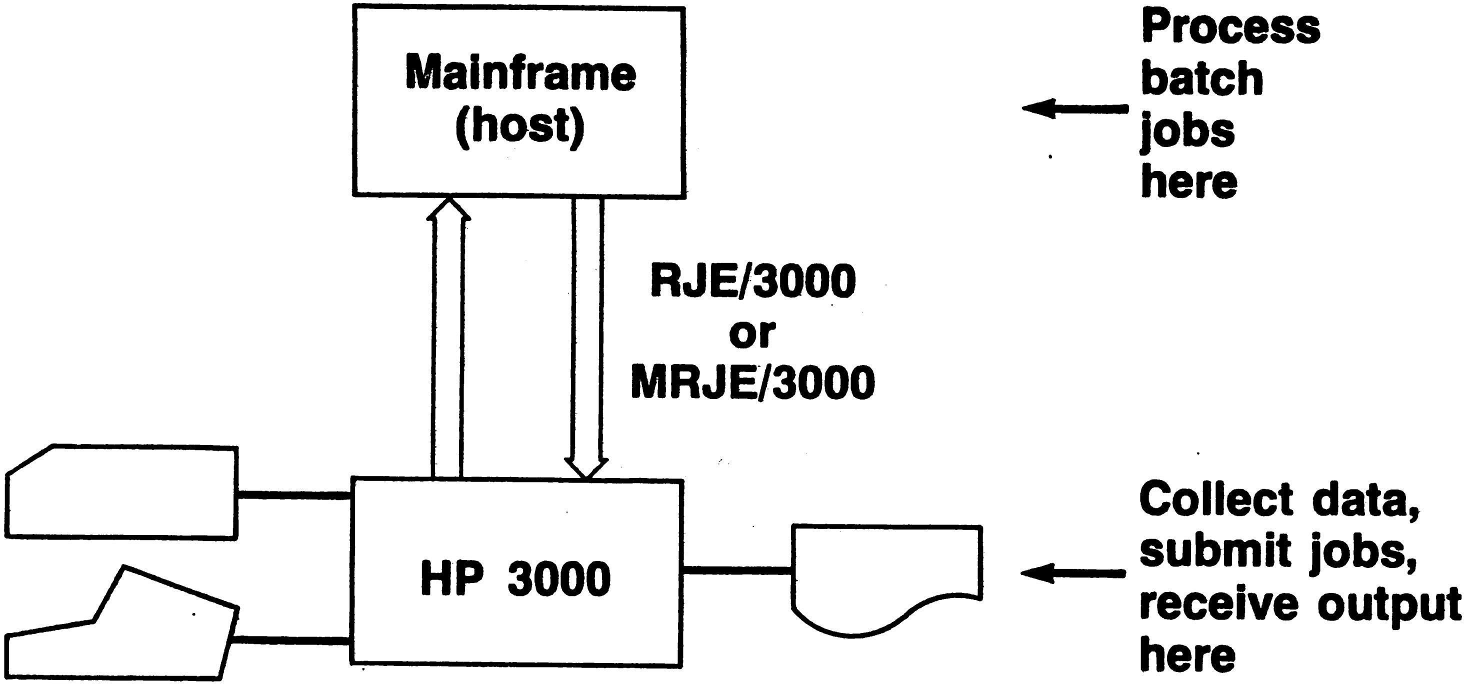
RJE/3000

Remote Job Entry
(2780/3780 emulator)

MRJE/3000

Multileaving Remote Job Entry

RJE and MRJE



Mainframe (host)

RJE/3000
or
MRJE/3000

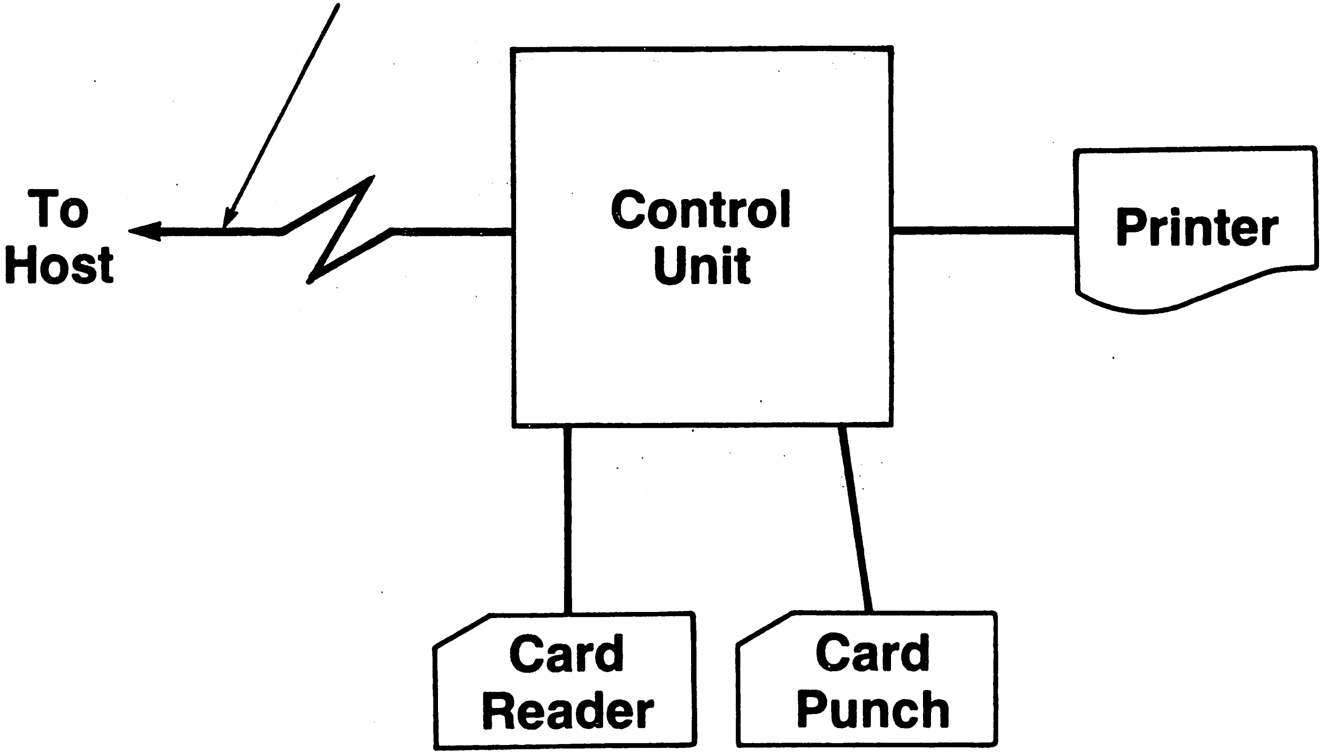
HP 3000

Process
batch
jobs
here

Collect data,
submit jobs,
receive output
here

IBM 2780/3780: NON-MULTILEAVING

Only one job stream on communication line at a time



To Host

Control Unit

Printer

Card Reader

Card Punch

IBM 2780

① Introduced 1967

② Non-multileaving

③ Used bisync (BSC) protocol

④ No CRT — punched cards provide control

IBM 3780

① Introduced 1972

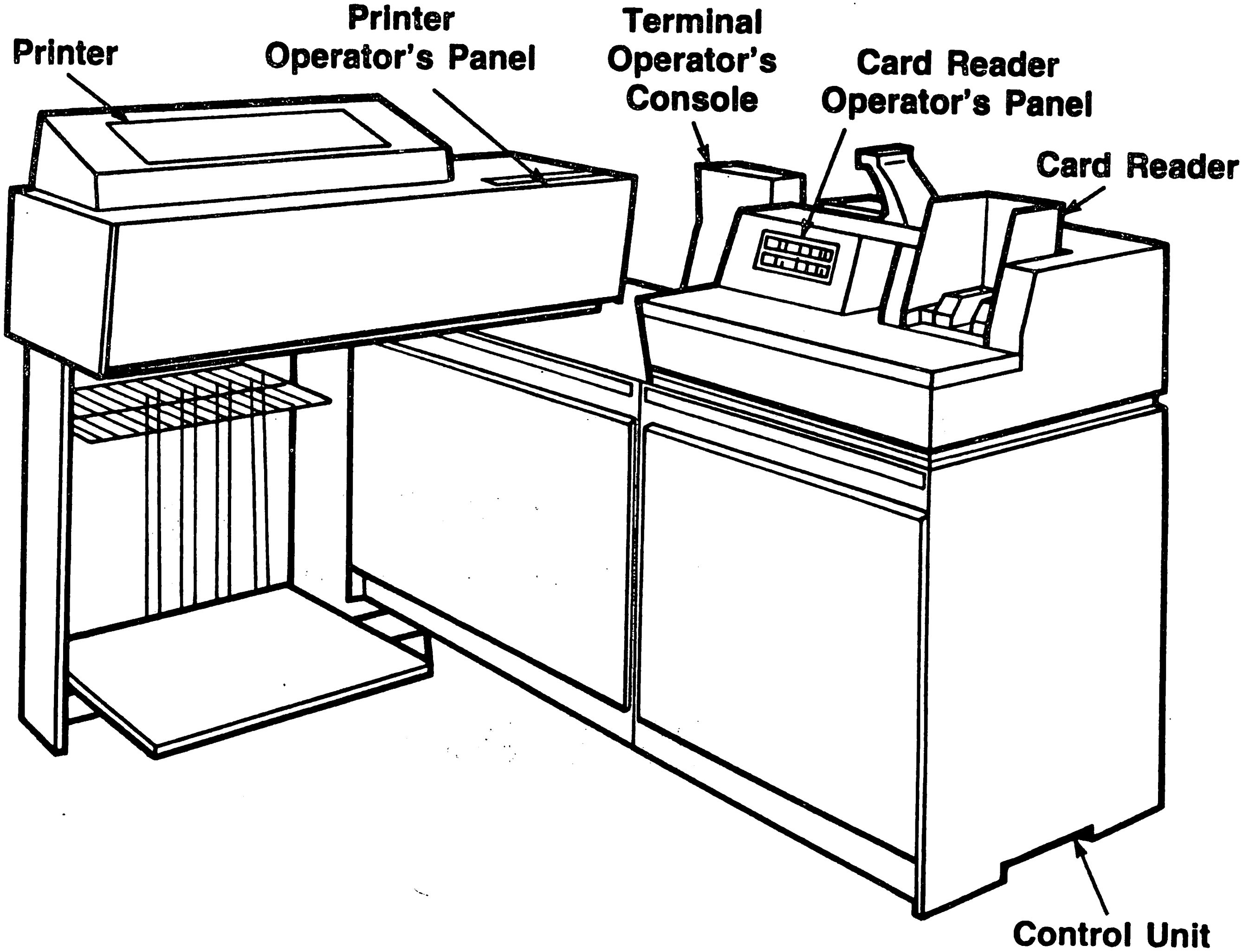
② Non-multileaving

③ Better performance than 2780:

- Faster data transmission
- 3780 compresses blanks
  (2780 has no data compression)
- Higher-speed devices

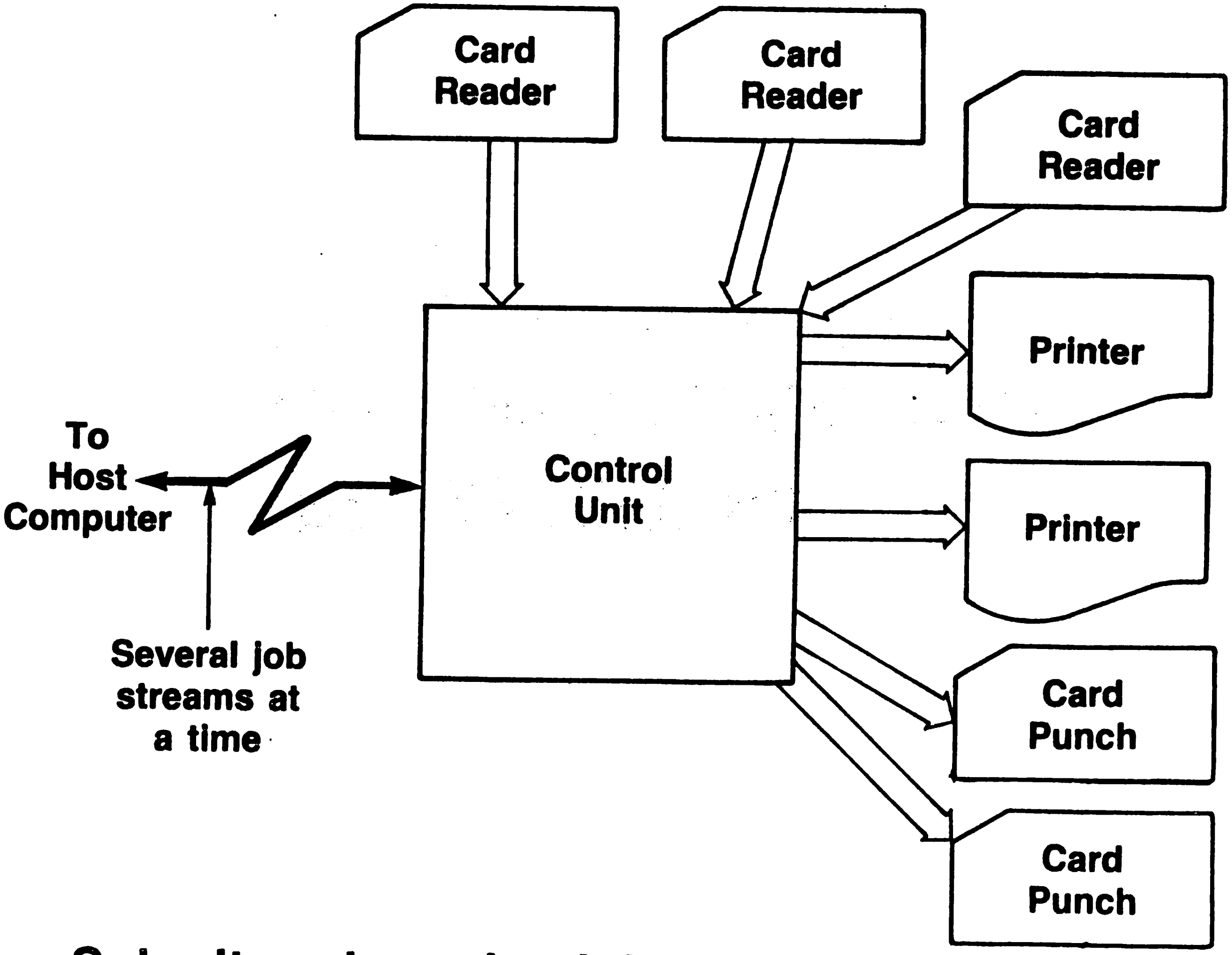
2780 + Better Performance = 3780

IBM 2780/3780


Printer
Printer Operator's Panel
Terminal Operator's Console
Card Reader Operator's Panel
Card Reader
Control Unit

IBM 2780/3780, AND MULTILEAVING WORKSTATIONS

MULTILEAVING WORKSTATION


Card Reader
Card Reader
Card Reader
Printer
Printer
Card Punch
Card Punch
Control Unit
To Host Computer
Several job streams at a time

Submit and receive jobs at the same time

IBM 360 MODEL 20

(360/20)

① Original multileaving workstation

② Smallest system of the IBM 360 series

③ Called a "HASP Workstation"
(a misnomer)

④ IBM 360/30 is similar to 360/20

A COMPARISON

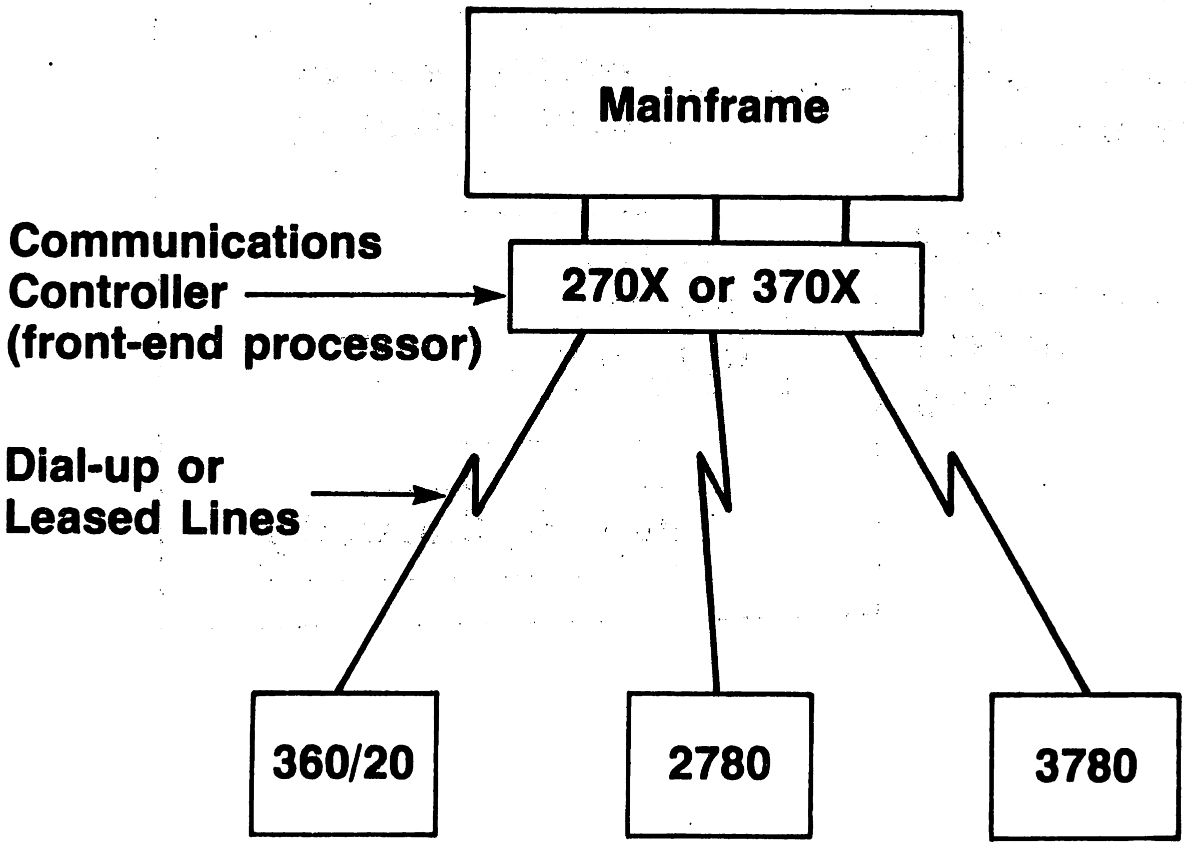
IBM 2780/3780

- *Non*-multileaving
- Handles:
    - 1 Printer
    - 1 Card Reader (opt)
    - 1 Card Punch (opt)
- No console
- 2780 — no character compression
  3780 — compresses blanks

IBM 360/20

- Multileaving
- Handles:
    - Several printers, card readers, card punches
- *May* have command console
- Full character compression

CONNECTING TO THE HOST



Mainframe

Communications Controller (front-end processor)

270X or 370X

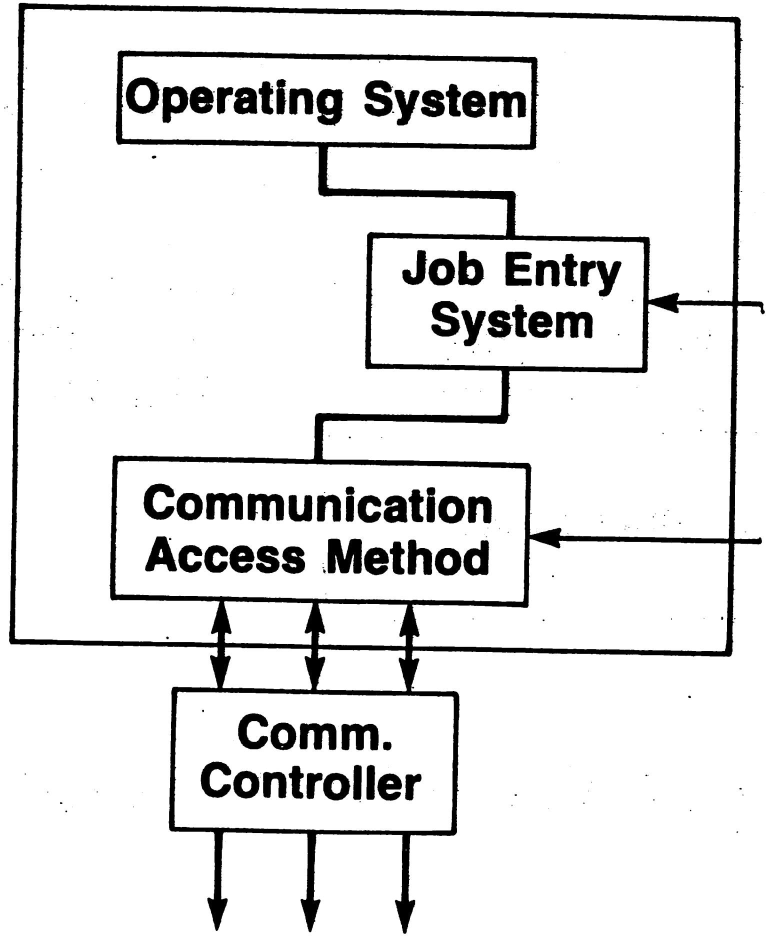
Dial-up or Leased Lines

360/20

2780

3780

REQUIRED HOST SOFTWARE


Operating System
Job Entry System
Communication Access Method
Comm. Controller

Spools, executes, and returns jobs

Interfaces with communication lines

JOB ENTRY SUBSYSTEMS

Receive, spool, execute, and return batch jobs

JES/RES

RSCS

JES 3

HASP II

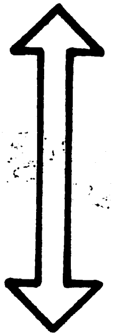
JES 2

ASP

POWER

COMMUNICATION ACCESS METHODS

Less Sophisticated



More Sophisticated

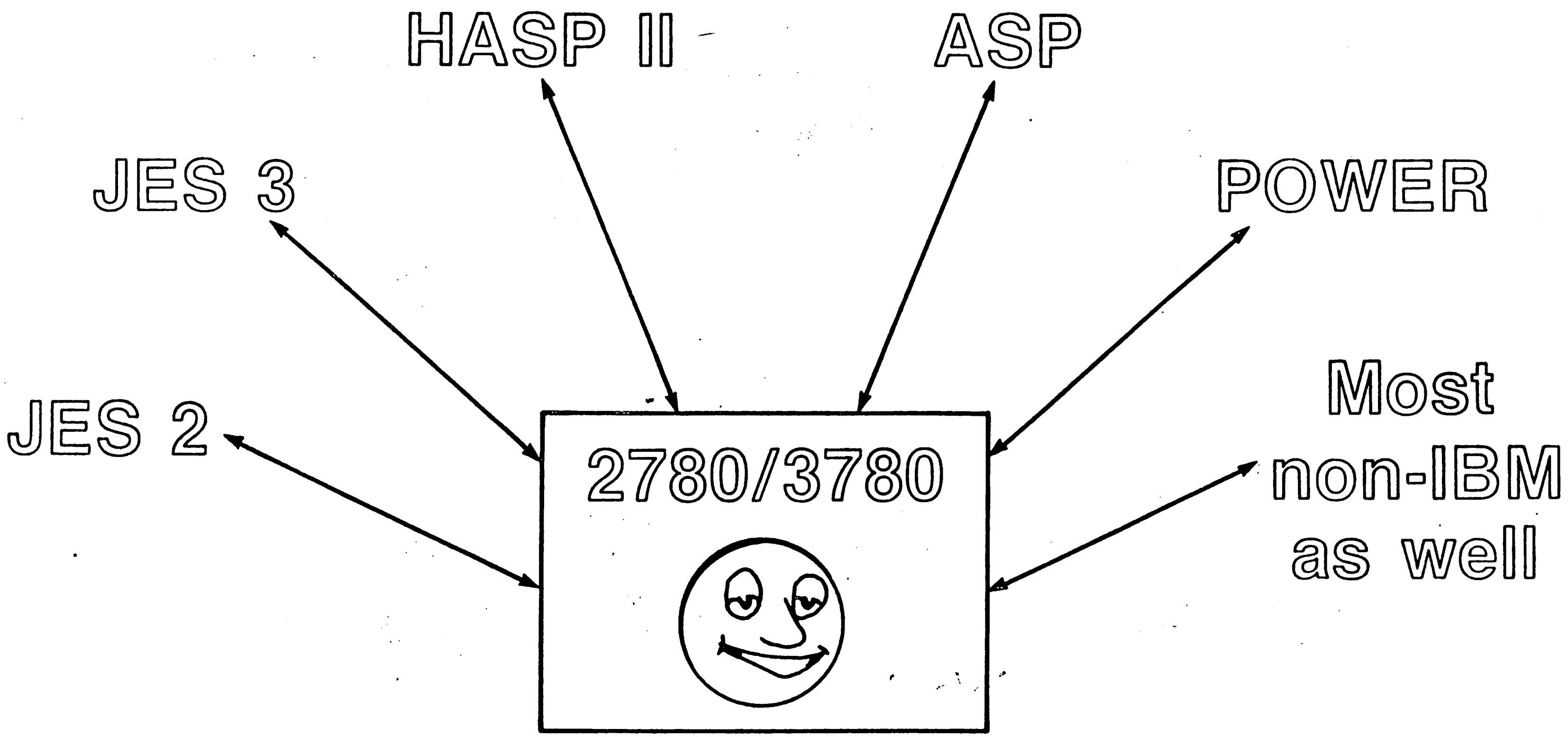
BTAM — Basic Telecommunication Access Method

TCAM — Telecommunication Access Method

VTAM — Virtual Telecommunication Access Method

RTAM — Remote Telecommunication Access Method
(RTAM comes built-into HASP)

2780/3780'S: COMPATIBLE WITH ESSENTIALLY ANY JOB ENTRY SYSTEM



HASP II

ASP

JES 3

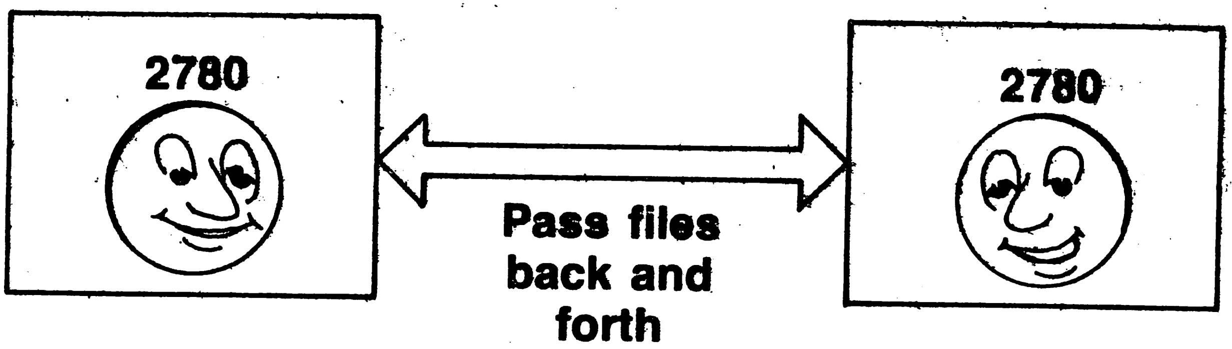
POWER

JES 2

2780/3780

Most non-IBM as well

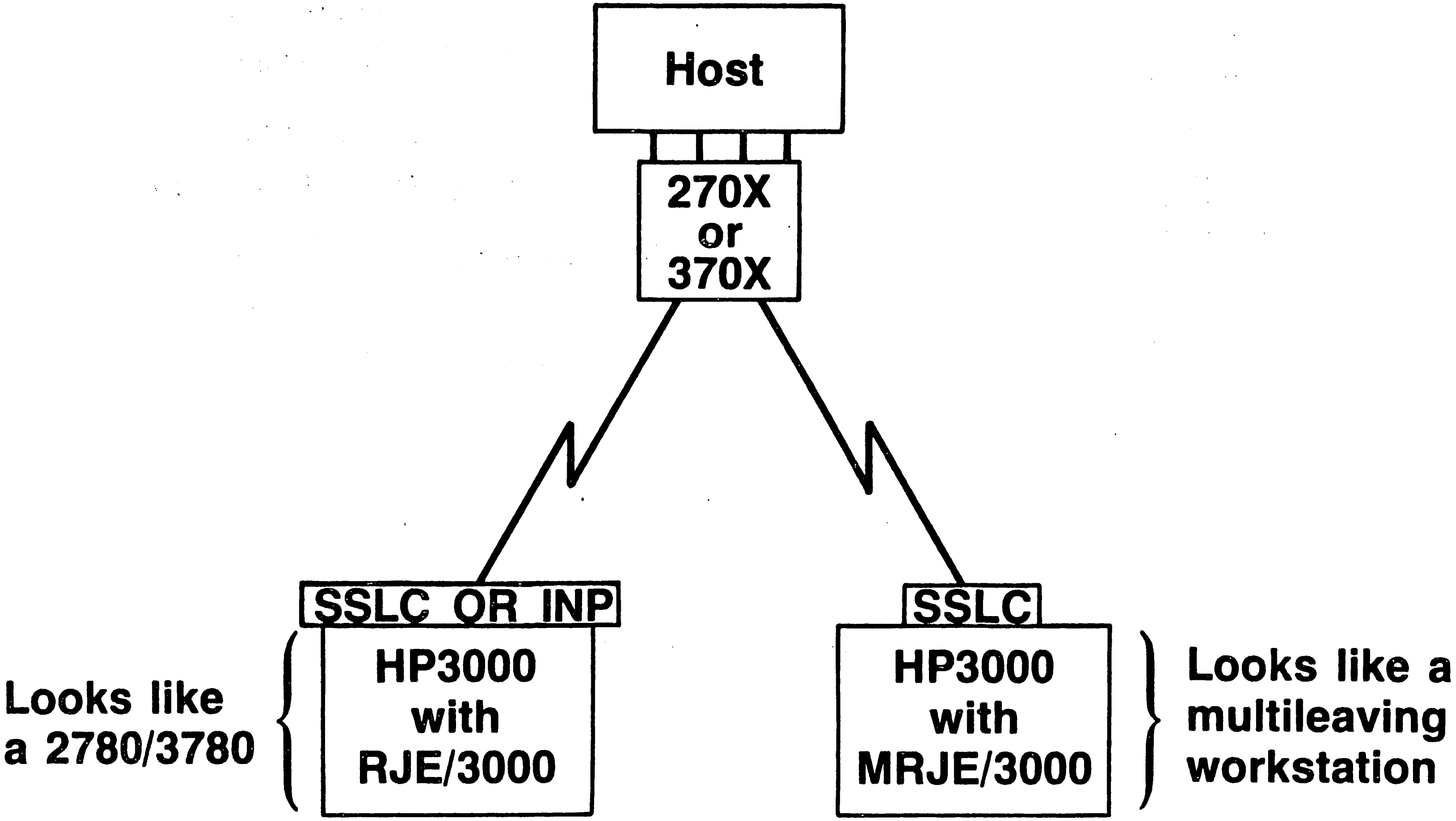
2780/3780'S CAN COMMUNICATE AMONG THEMSELVES



Pass files back and forth

Note: "Communication" ≠ Remote Job Entry

CONFIGURATION OF RJE/3000 OR MRJE/3000


Host
270X
or
370X
SSLC OR INP
HP3000
with
RJE/3000
Looks like
a 2780/3780
SSLC
HP3000
with
MRJE/3000
Looks like a
multileaving
workstation

COMPATIBILITY WITH OTHER SYSTEMS

RJE/3000

1) Any IBM mainframe
2) Most non-IBM mainframes
3) Most systems that emulate or support 2780/3780's

MRJE/3000

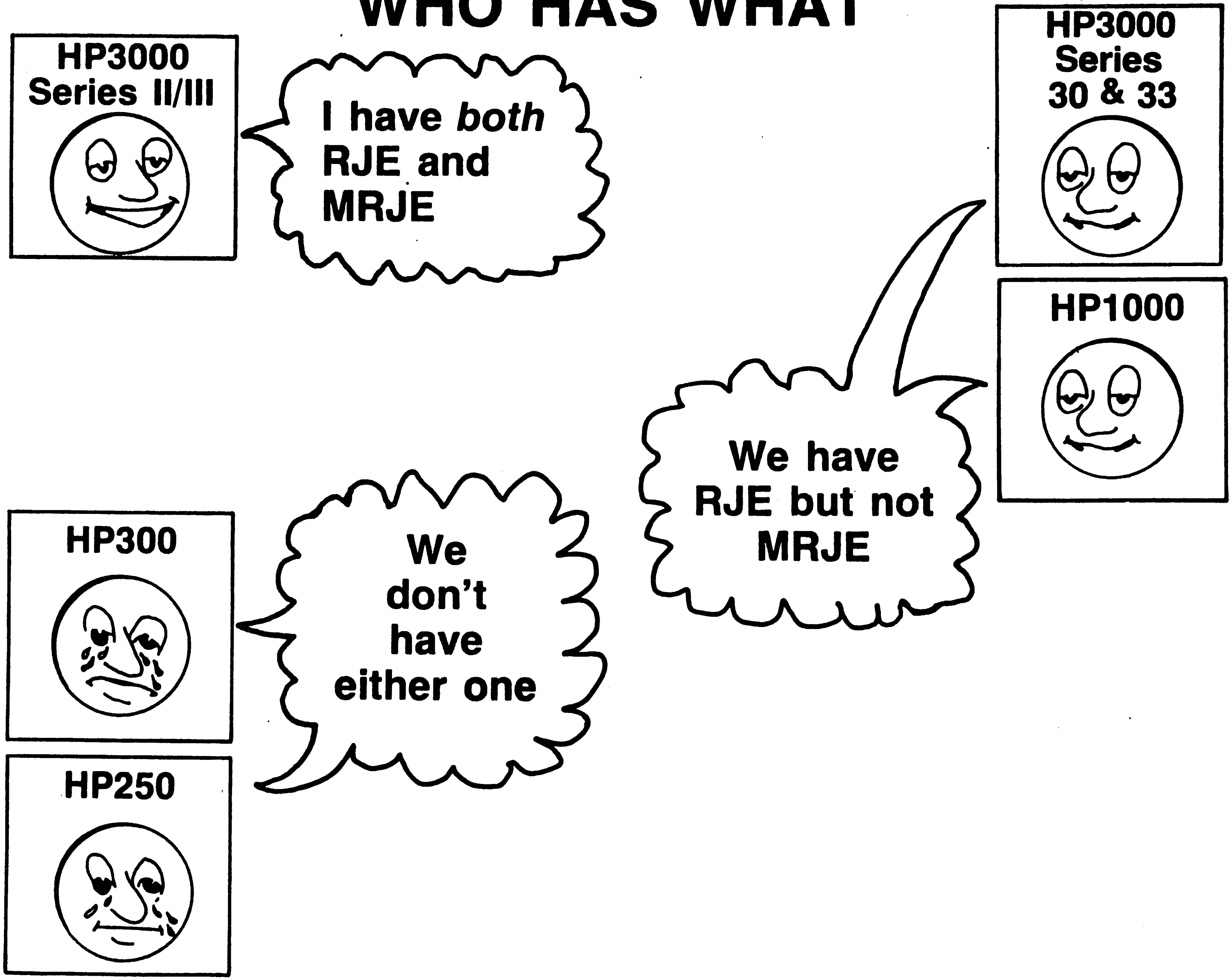
HASP II,
JES 2,
JES 3,
or ASP
only

RJE/3000 HAS BEEN USED WITH:

Hosts	Non-Hosts
Burroughs 6800	IBM System/3
Control Data	Data 100's
Honeywell	DEC PDP-11

Your SE should always test from HP sales office first

WHO HAS WHAT



HP3000 Series II/III

I have *both* RJE and MRJE

HP3000 Series 30 & 33

HP1000

We have RJE but not MRJE

HP300

HP250

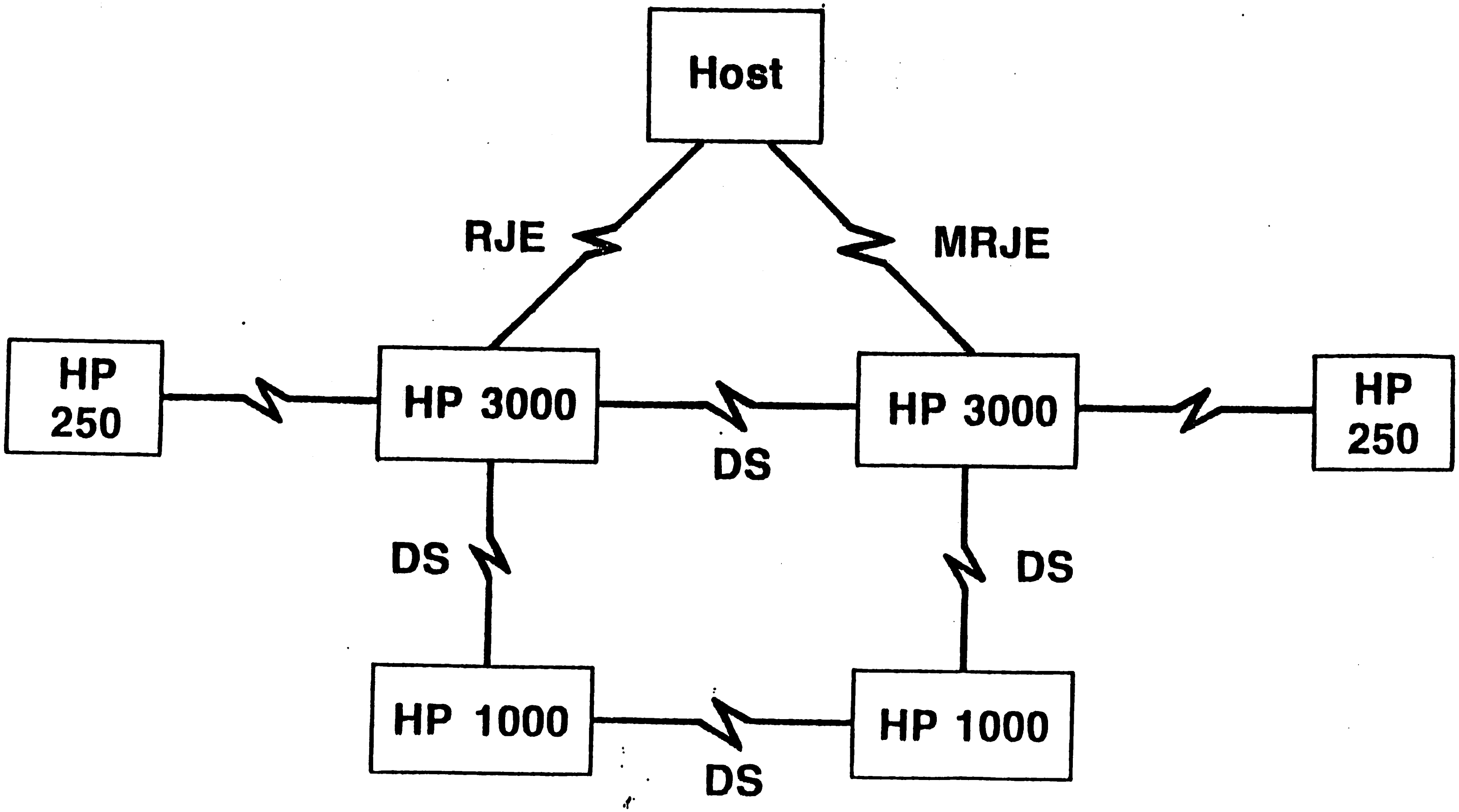
We don't have either one

SUMMARY OF MAIN POINTS

- MRJE is more capable than RJE
  - Multiple users
  - Job output management
  - MRJE manager
  - Joblog file
  - Submit jobs on-line or off-line
- MRJE is more restricted than RJE

  MRJE — HASP II, JES 2, JES 3, and ASP only
  RJE — Any 2780/3780 system
- MRJE has higher overhead than RJE

RJE AND MRJE: PART OF HP — DSN



Host

RJE

MRJE

HP 250

HP 3000

DS

HP 3000

HP 250

DS

DS

HP 1000

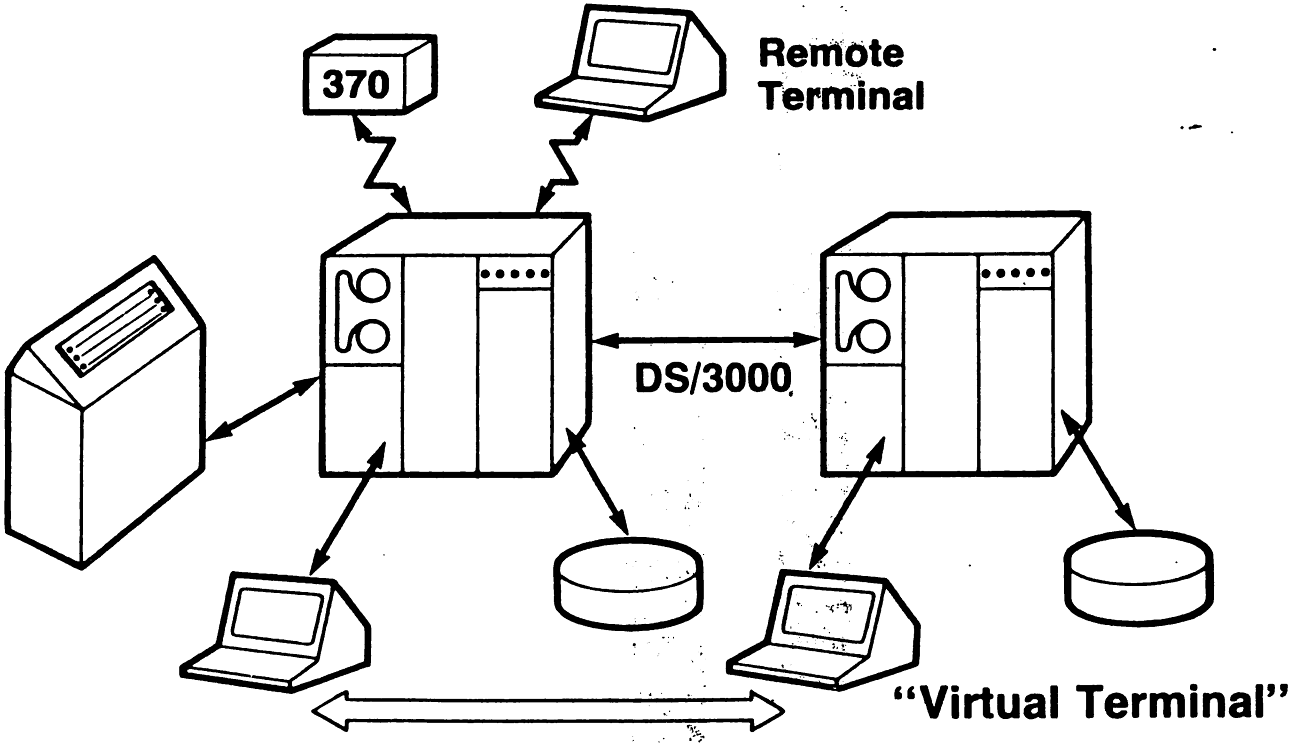
HP 1000

DS

HP — Distributed Systems Network

HP'S DISTRIBUTED NETWORK

HJ 2

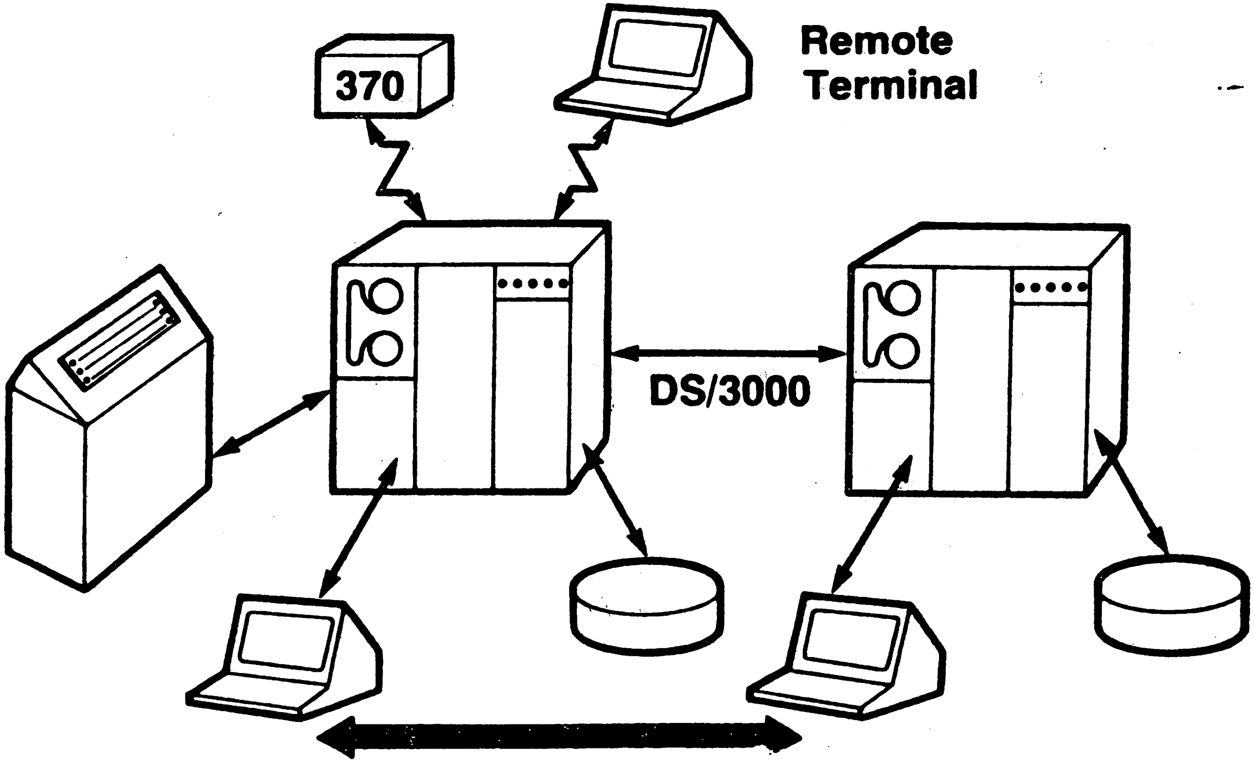

370
Remote Terminal
DS/3000
"Virtual Terminal"

Remote Commands

Run all compilers & subsystems

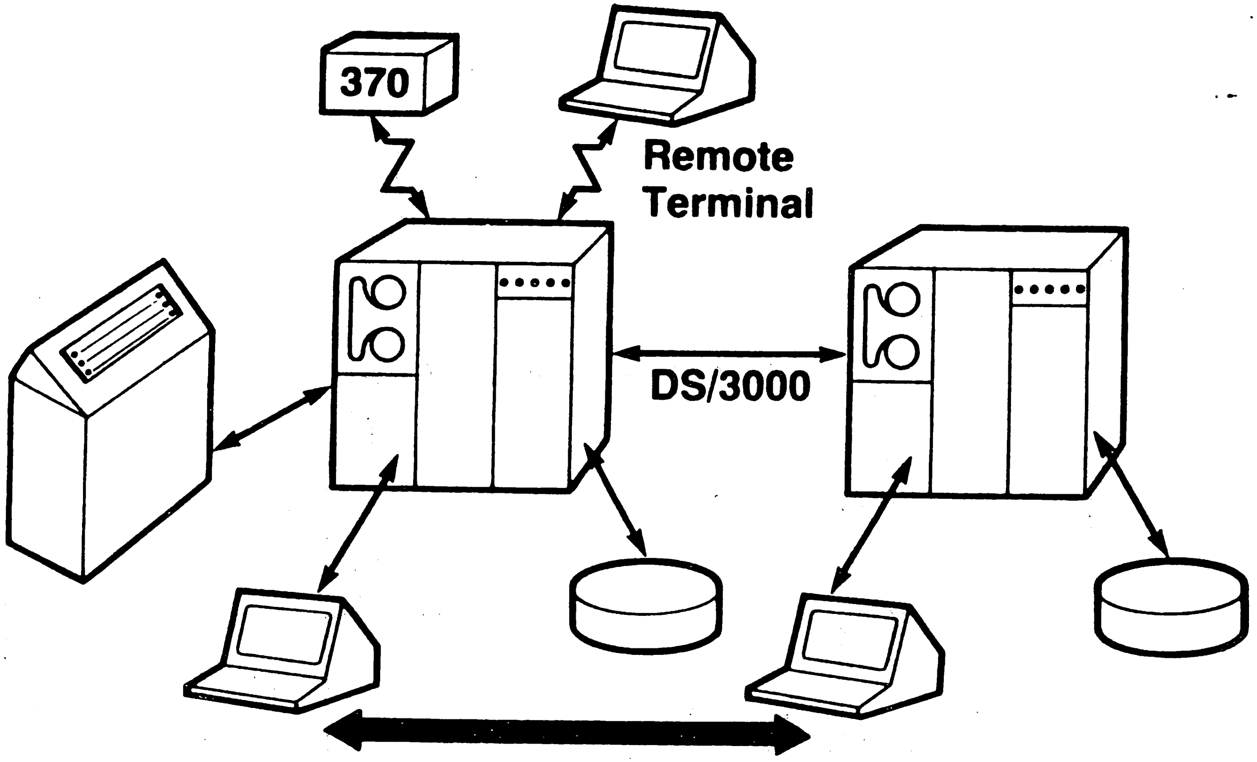
SPL, COBOL, FORTRAN, RPG, BASIC, APL, LISTF, STREAM, RESTORE, SAVE, BUILD, PURGE, PREP, use of peripherals, acct/system manager, system supervisor

RFA


370
Remote Terminal
DS/3000

FCHECK
FCLOSE
FCONTROL
FGETINFO
FUNLOCK
FOPEN
FPOINT
FREADDIR
FREADSEEK
FREAD
FRELATE
FRENAME
FSPACE
FSETMODE
FUPDATE
FREADLABEL
FWRITE
FWRITEDIR

PTOP


370
Remote Terminal
DS/3000

Master

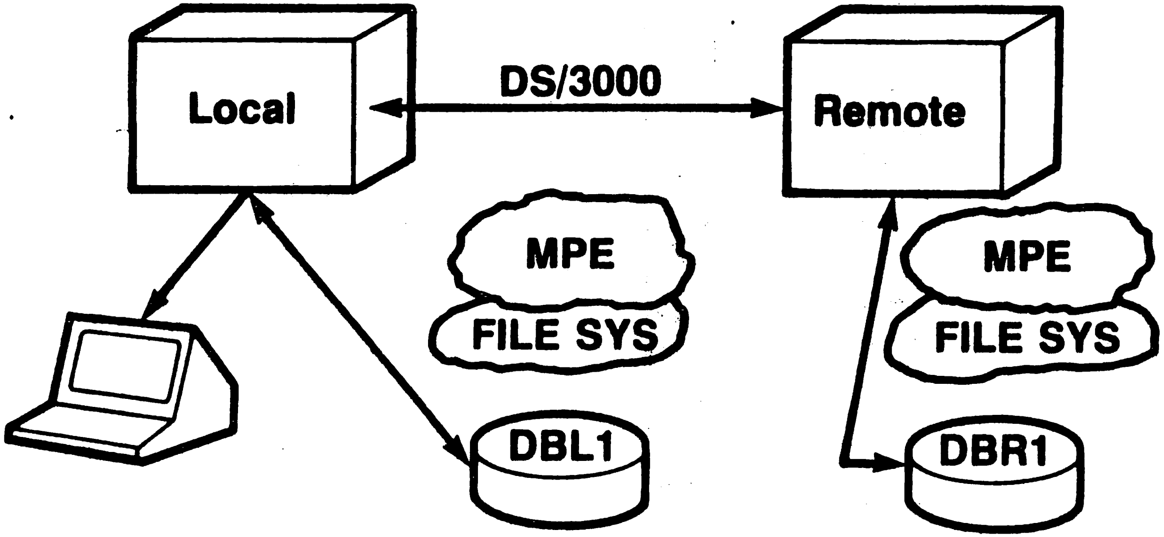
POPEN
PWRITE
PCLOSE

PREAD
PCONTROL
PCHECK

Slave

GET
ACEPT
REJECT

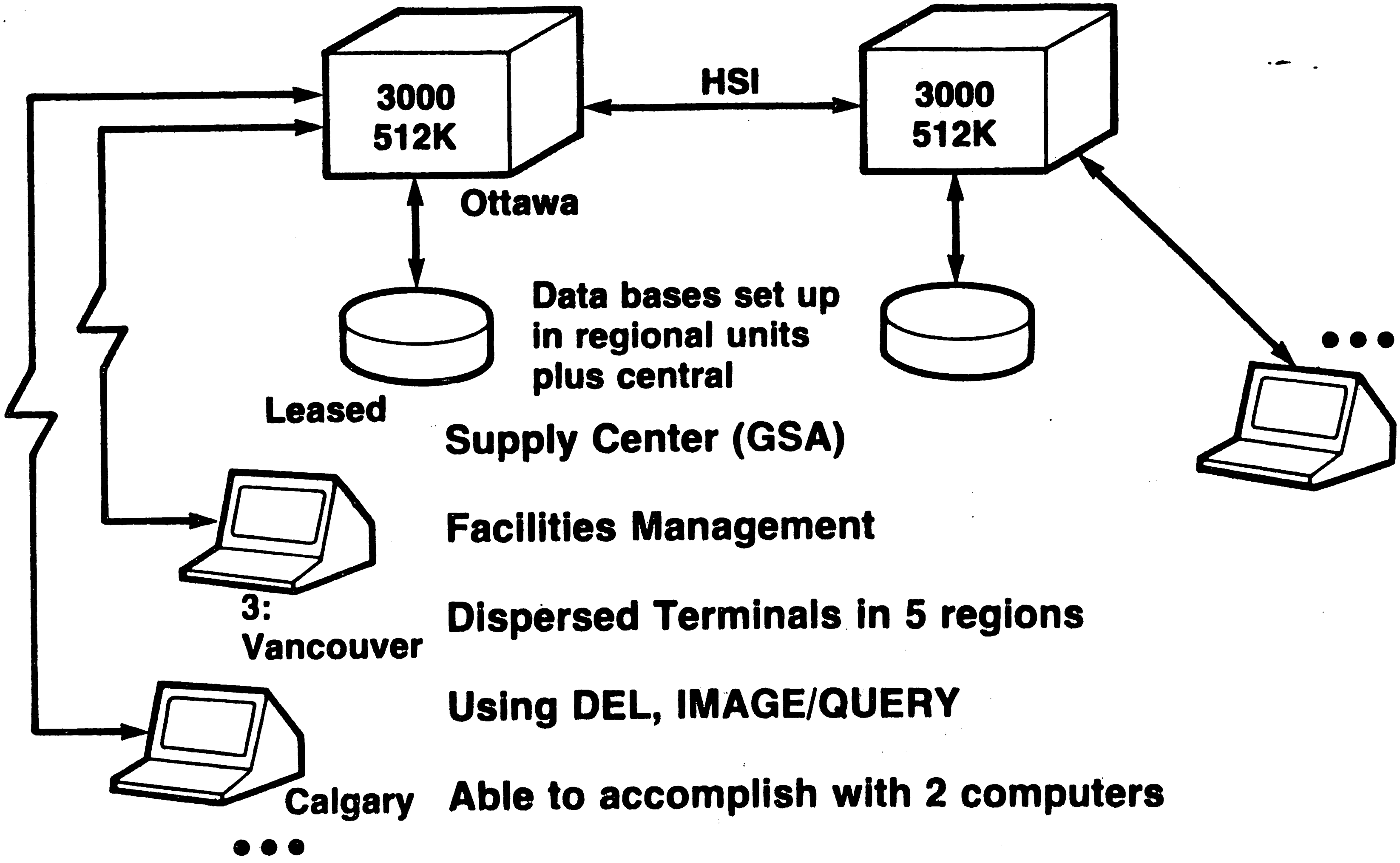
Remote
Data Base Access


Local
DS/3000
Remote
MPE
FILE SYS
DBL1
MPE
FILE SYS
DBR1

DS–NETWORK–EXAMPLES

HJ 3

DS Example (1)


3000
512K
HSI
3000
512K
Ottawa
Leased
3:
Vancouver
Calgary

Data bases set up in regional units plus central

Supply Center (GSA)

Facilities Management

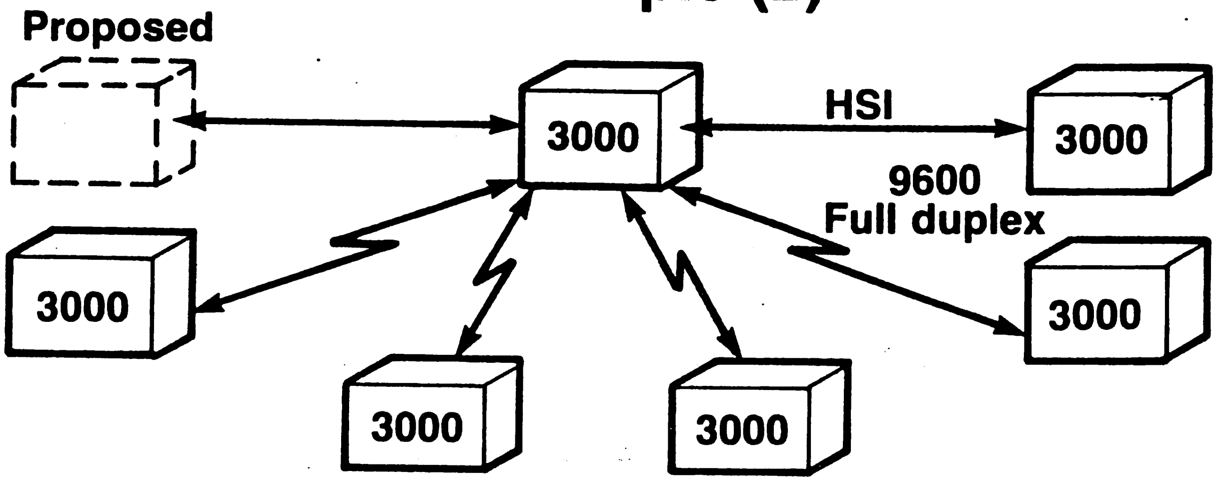
Dispersed Terminals in 5 regions

Using DEL, IMAGE/QUERY

Able to accomplish with 2 computers

Original plans were to disperse 5 units geographically

DS Example (2)


Proposed
3000
HSI
3000
9600
Full duplex
3000
3000
3000
3000
3000

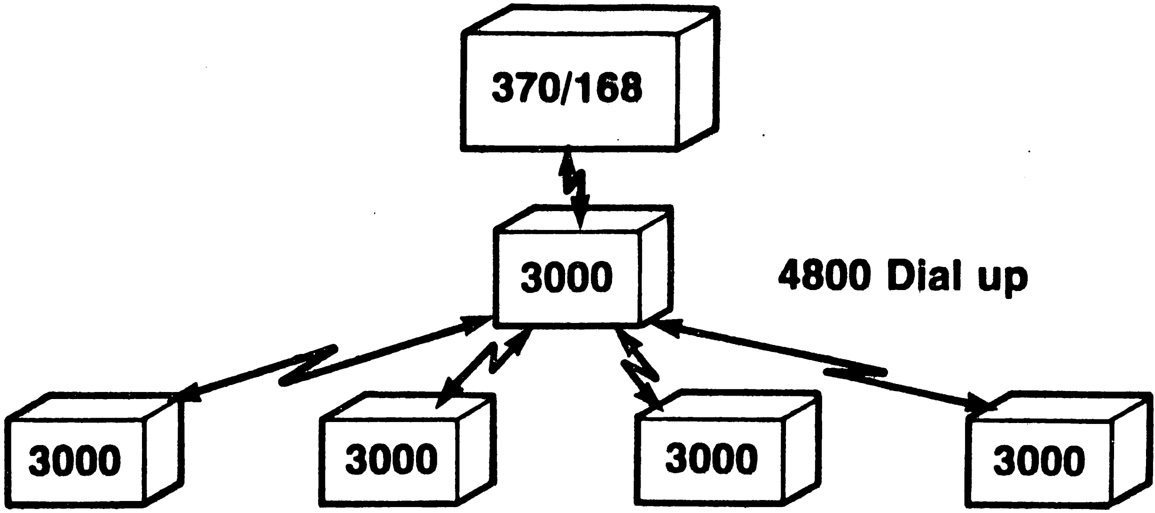
Remote 3000's used for high school instructional purposes, and payroll

DS Requirements (for each system)

Daily	100 records	256 bytes
	50 records	275 bytes
2 weeks	3000 records	24 bytes
	12000 records	220 bytes

When using FCOPY dissatisfied with performance

DS Example (3)


370/168

3000

4800 Dial up

3000

3000

3000

3000

Corporate data transfer

- 17 3000's dispersed geographically

Transfer profit/loss accounting data to corporate headquarters

Central EDP group

- Do all programming
- Other 3000's don't need compilers
- Use UDC (user defined commands)

Auto dial capability — special uses:

- Accounting data
- Transfer programs & files
- Electronic mail

What DS/3000 Can't Do

Store & forward

- The Data Buffer does not contain address information

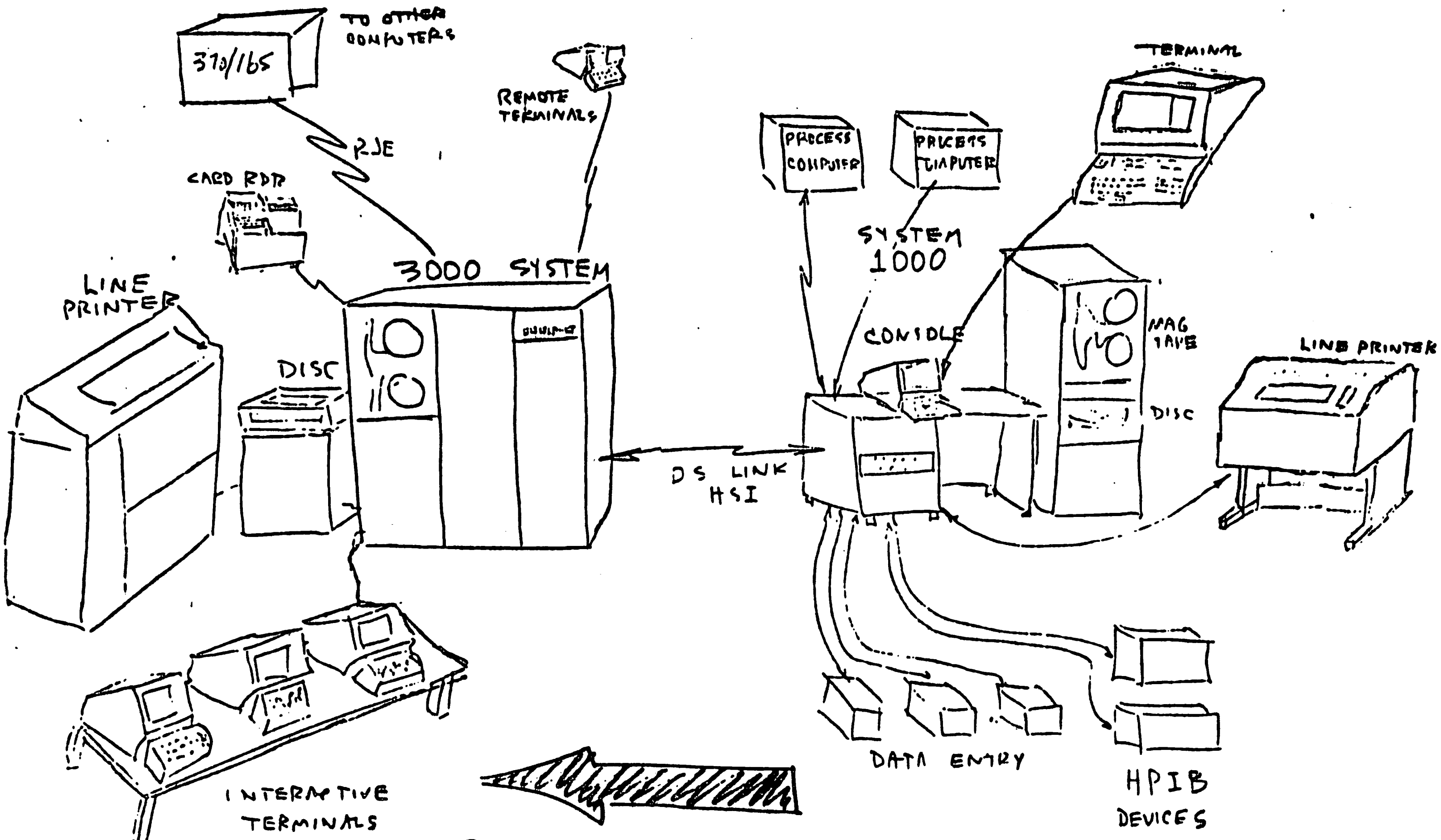
Automatic routing through network

- Routing must be specified with the FILE equation or with PTOP programs at each node

Autodialing

- Coded by applications using an ATC port

Nodal addressing


TO OTHER COMPUTERS
370/165
RJE
REMOTE TERMINALS
CARD RDR
LINE PRINTER
DISC
3000 SYSTEM
PROCESS COMPUTER
PROCESS COMPUTER
SYSTEM 1000
TERMINAL
CONSOLE
MAG TAPE
DISC
LINE PRINTER
DS LINK HSI
INTERACTIVE TERMINALS
DATA ENTRY
HPIB DEVICES

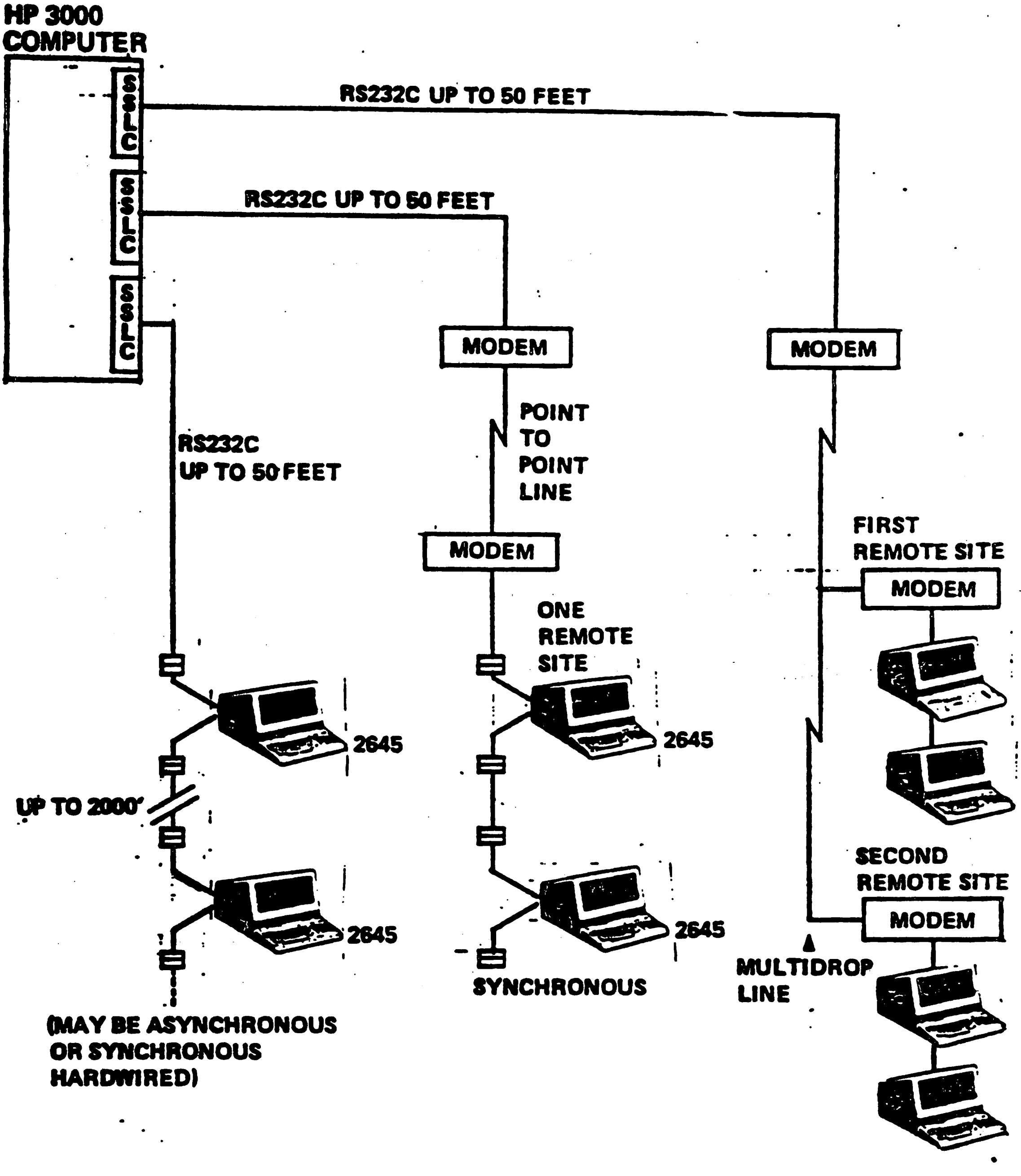
REMOTE COMMANDS

RUN ALL COMPILIERS/SUBSYSTEMS
SPL, COBOL, FORTRAN, RPG, BASIC, APL
LISTF, STREAM, RESTORE, SAVE
BUILD, PURGE, PREP
ACCT/SYSTEM MANAGER, SYSTEM SUPERVISOR
PROVIDES USE OF PERIPHERALS

TERMINAL-TO-COMPUTER COMMUNICATION

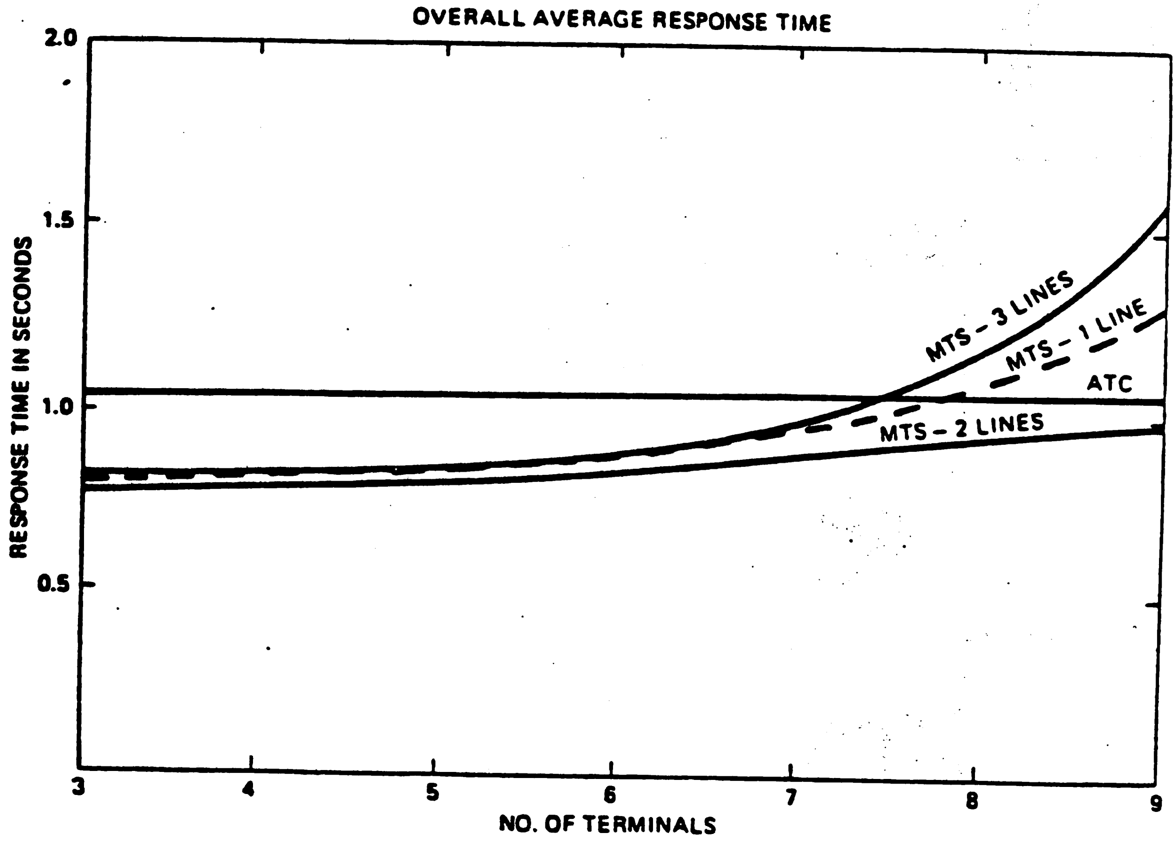
- **ATC - ASYNCHRONOUS CONTROLLER**
- **MTS - MULTIPOINT**

HP 3000
MTS/3000 TERMINAL CONFIGURATIONS


HP 3000
COMPUTER
SSLC
SSLC
SSLC
RS232C UP TO 50 FEET
RS232C UP TO 50 FEET
RS232C
UP TO 50 FEET
MODEM
MODEM
POINT
TO
POINT
LINE
MODEM
FIRST
REMOTE SITE
MODEM
ONE
REMOTE
SITE
2645
2645
UP TO 2000'
2645
2645
SECOND
REMOTE SITE
MODEM
MULTIDROP
LINE
SYNCHRONOUS
(MAY BE ASYNCHRONOUS
OR SYNCHRONOUS
HARDWIRED)

NOTE: HARDWIRED CONFIGURATIONS WITH THE ASYNCHRONOUS REPEATER (HP 30037A) ARE NOT SHOWN.

ATC VERSUS MTS/3000



Control Sequences

- **Polling — An "Invitation to send a message"**

  EOT PAD AA " " ENQ PAD

- **Selection — A "Command to receive a message"**

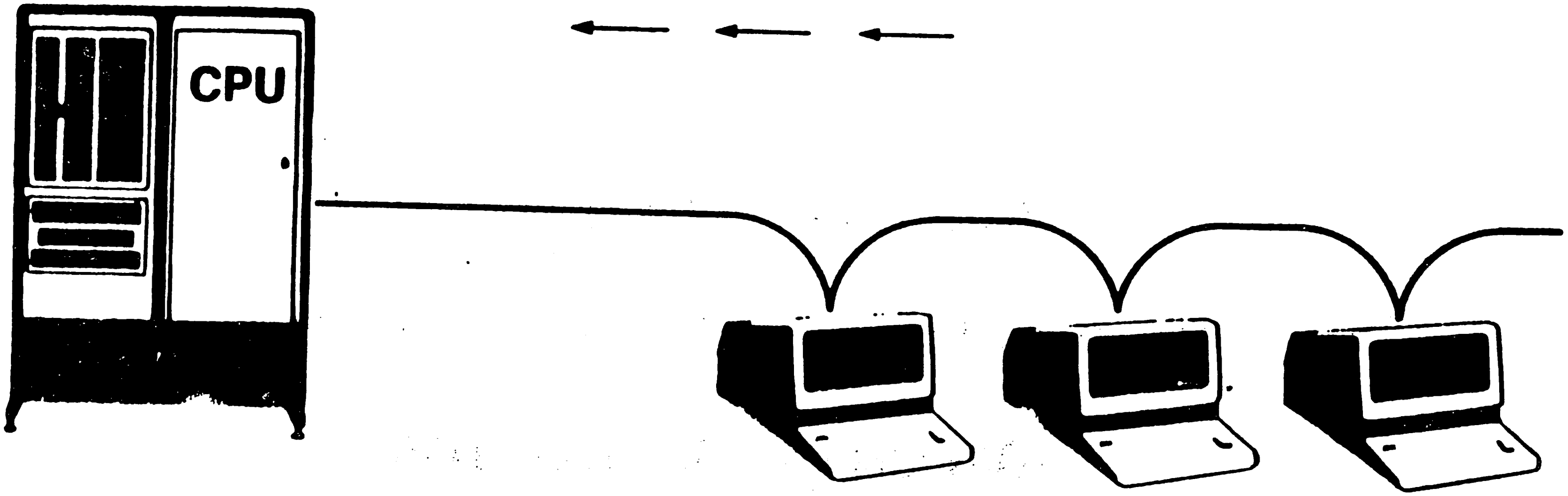
  EOT PAD aa BB ENQ PAD

Poll: Do You Have Anything To Send Me?

→ → → → →

Yes or No

← ← ←



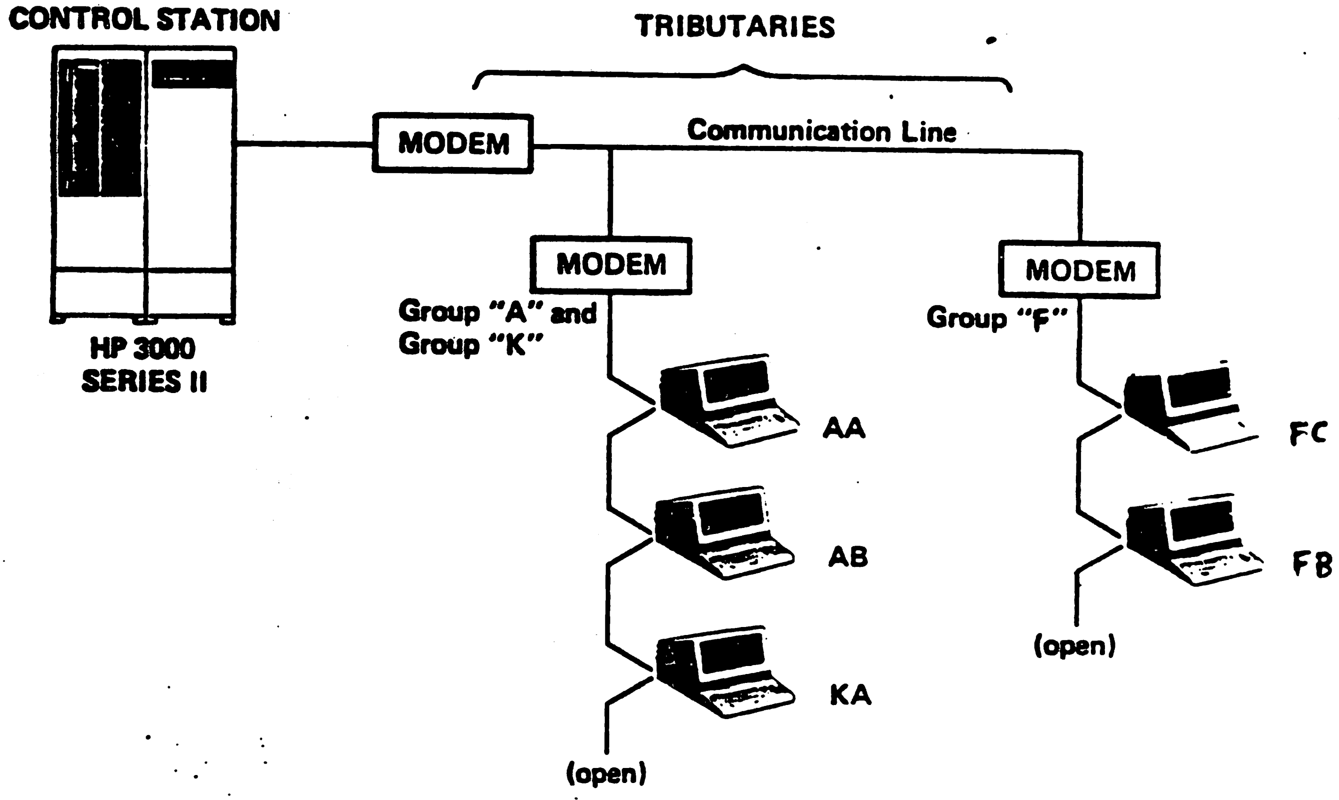
Select: May I Send You Something?

→ → → → →

Yes or No

← ← ←

MTS/3000 TERMINAL ADDRESSING

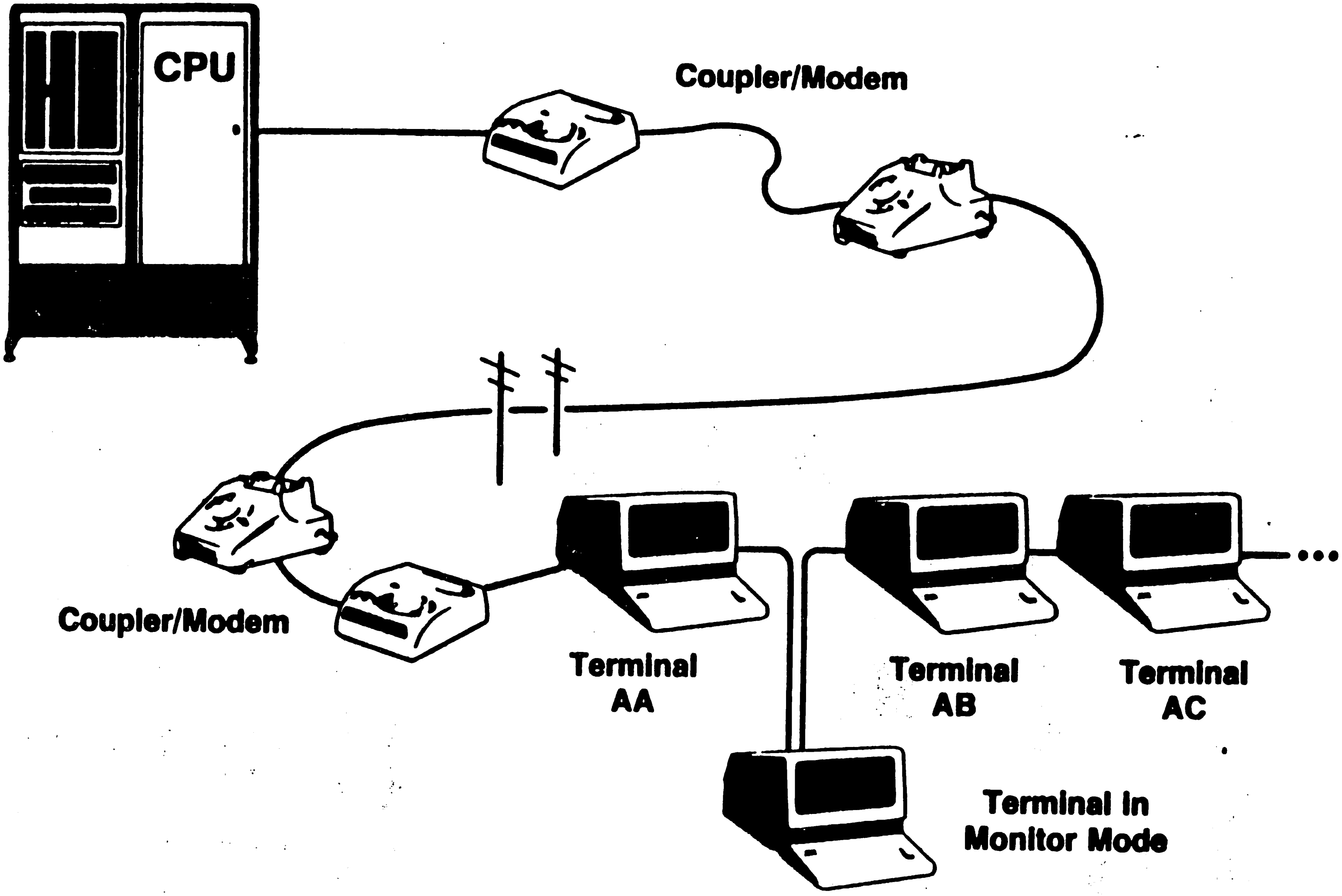

CONTROL STATION
TRIBUTARIES
MODEM
Communication Line
HP 3000
SERIES II
MODEM
Group "A" and
Group "K"
MODEM
Group "F"
AA
AB
KA
(open)
FC
FB
(open)

Synchronous Polling Addressing

```
<SYN><EOT><PAD><SYN><GROUP ID><GROUP ID><DEV ID><DEV ID><ENQ><PAD>

SSS   E   P   SSS      A       A       "       "      E    P
YYY   O   A   YYY                                     N    A
NNN   T   D   NNN                                     Q    D
```

Synchronous Selection Addressing

```
<SYN><EOT><PAD><SYN>><GROUP ID><GROUP ID><DEV ID><DEV ID><ENQ><PAD>

SSS   E   P   SSS      a       a       B       B      E    P
YYY   O   A   YYY                                     N    A
NNN   T   D   NNN                                     Q    D
```


CPU
Coupler/Modem
Coupler/Modem
Terminal AA
Terminal AB
Terminal AC
Terminal in Monitor Mode

B C C

GOOD

ACK 1/0 — Acknowledge

WACK — Wait before transmit

RVI — Reverse interrupt

BAD

NAK — Negative acknowledge

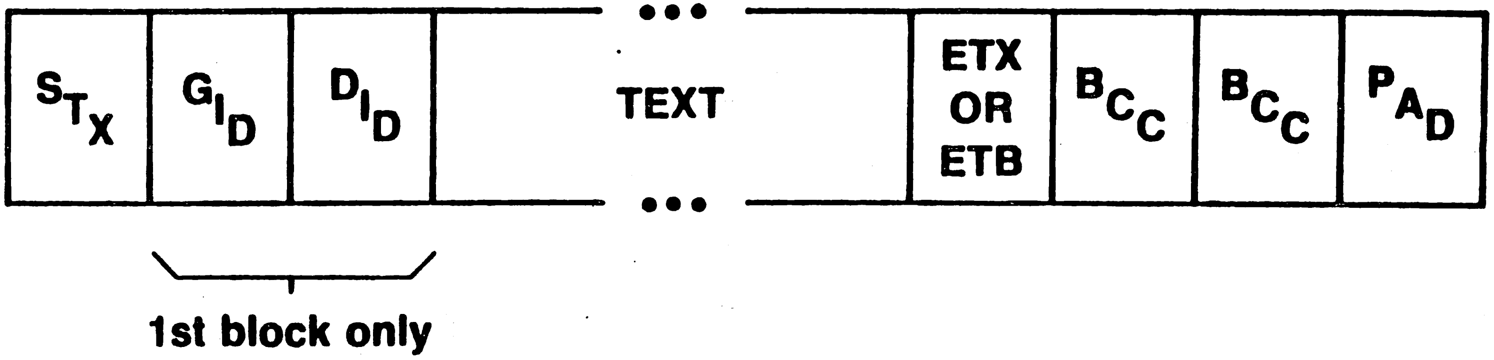
(**EOT** — Buffer overflow)

Block Check Characters

B C C

- Vertical Redundancy Check (VRC) — Parity
- Longitudinal Redundancy Check (LRC)
- Cyclic Redundancy Check (CRC 16)

Text Block



STX	GID	DID	••• TEXT •••	ETX OR ETB	BCC	BCC	PAD

1st block only

Current Telecommunications Service Offering:

The three primary categories of telecommunications services available to domestic users in Europe are summarized in Figure 1:

- Switched network offerings have historically been the private domain of the PTT's in Europe and are likely to remain so for the forseeable future and this also includes the dedicated or leased line services. Both these traditional families of offerings may in a gross sense be regarded as data-transparent "bit-banging" types of services since the common carriers have traditionally been solely concerned with providing a virtually error free replica of the serial bit stream at the sink location.

MOREOVER, both of these types of offerings are generally concerned with providing a transmission environment that minizizes the absolute time delay in propagating bits or characters through the carrier networks.

Another characteristic of today's traditional offerings is that they tend to be protocol-independant to the extent that most services do not get more than superficially involved with the innards of user computers such as line control software, telecommunications access methods, buffering etc.

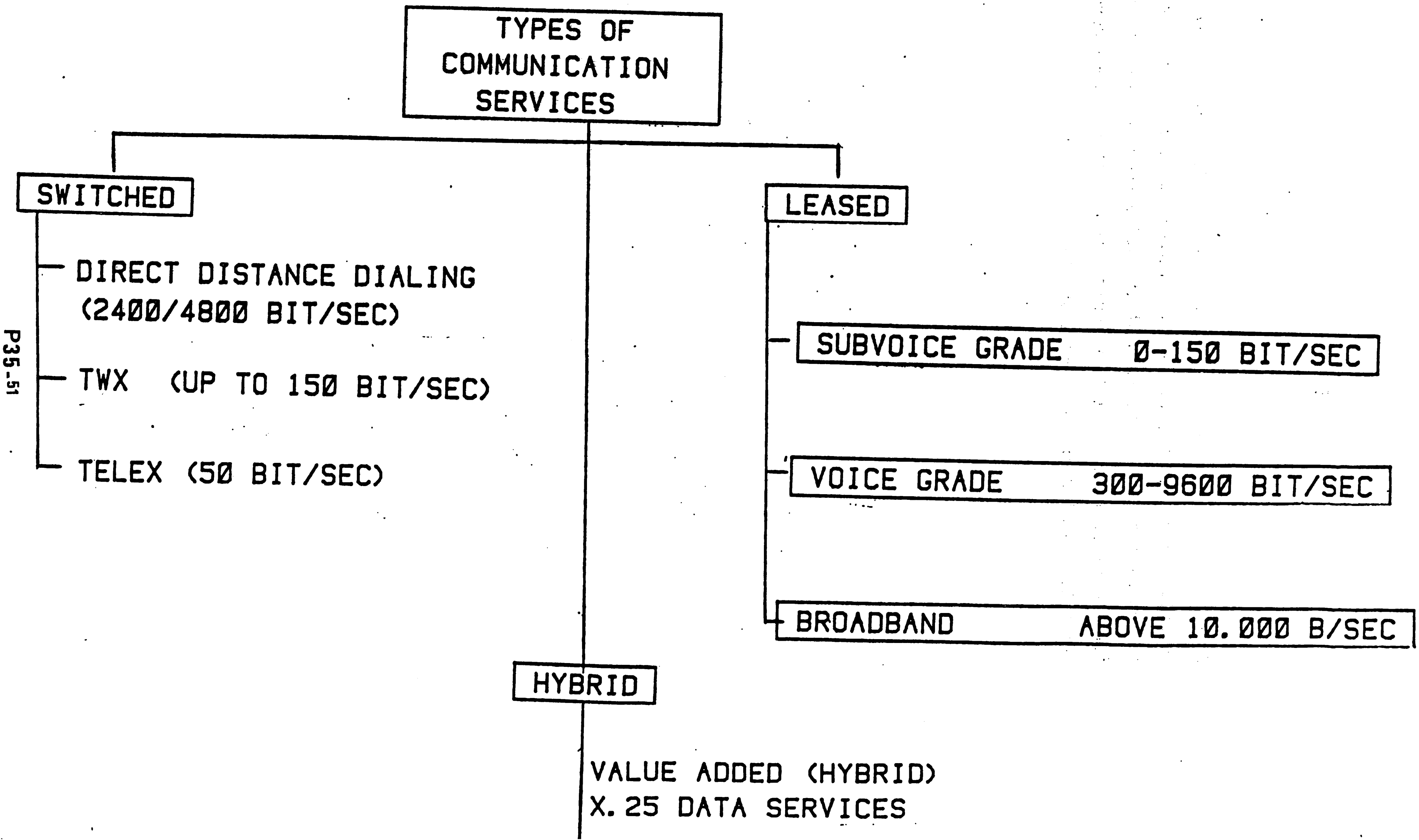
Insight into the potential communications requirements of computer networks is gained by noting that switched services are currently unavailable at bit rates faster then 4800 bits/sec. For leased lines the upper speed limit is about 48 Kbits in most of the European countries.

- HYBRID TELECOMMUNICATIONS SERVICE OFFERINGS

For the purpose of this discussion a hybrid data service offering here is one which has some characteristics (as seen by the user) of both switched and leased line offerings (This is not to be confused with a composite or hybrid data processing/data communications service in which both data processing and communications services are packaged into a single offering).

Also known as a "value added network service", it will provide an important new category of domestic common carrier service in which common carriers (in the form of the value added network vendors) for the first time in history depart from numerous time - honored concepts embodied in the traditionalservice categories.

CURRENT TELECOMMUNICATIONS SERVICE OFFERRINGS



TYPES OF COMMUNICATION SERVICES

- SWITCHED
  - DIRECT DISTANCE DIALING (2400/4800 BIT/SEC)
  - TWX (UP TO 150 BIT/SEC)
  - TELEX (50 BIT/SEC)
- HYBRID
  - VALUE ADDED (HYBRID) X.25 DATA SERVICES
- LEASED
  - SUBVOICE GRADE 0-150 BIT/SEC
  - VOICE GRADE 300-9600 BIT/SEC
  - BROADBAND ABOVE 10.000 B/SEC

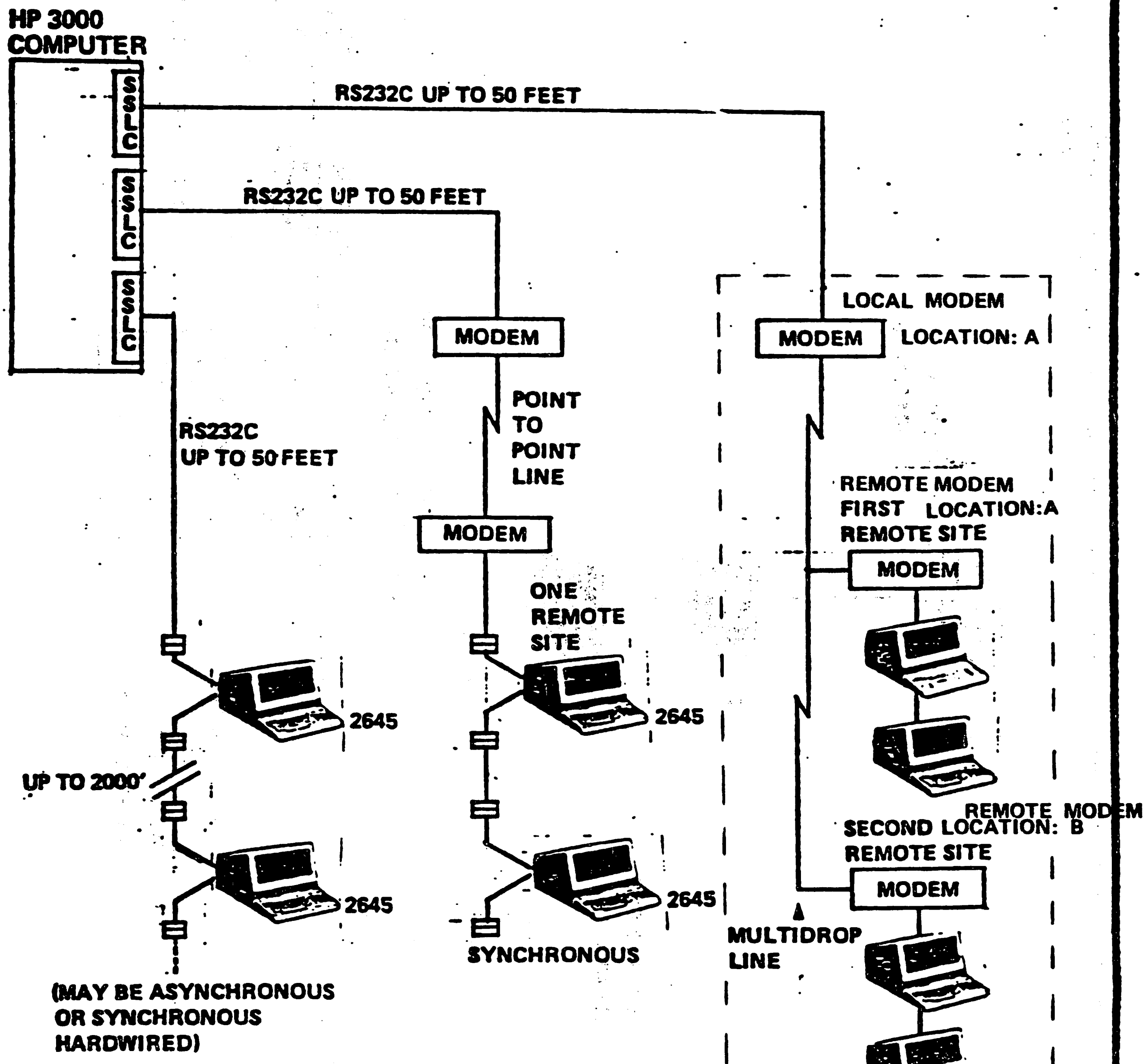
POTENTIAL PROBLEM AREAS

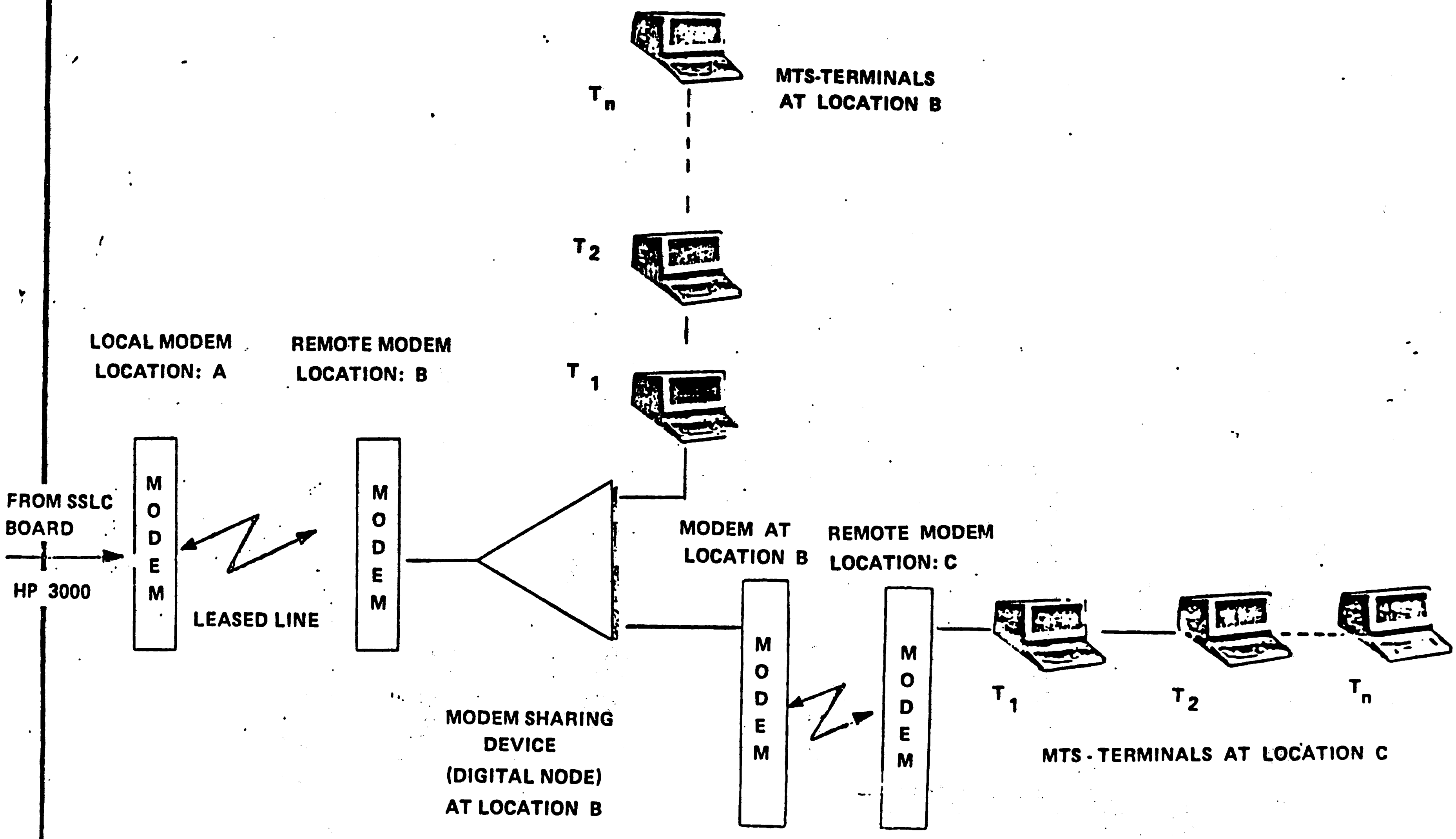
- APPLICATION (PERFORMANCE)
- MODEM
- PTT TELEPHONE NETWORKS
- CARRIERS AND REGULATORY AUTHORITIES

TYPE OF PROBLEMS

- DIFFERENCES IN V.24/RS 232 C INTERFACE
- DIFFERENCES IN MODULATION TECHNIQUES
- DIFFERENCES IN LINE CONNECTION- AND DISCONNECTION PROCEDURES
- DIFFERENCES IN LINE QUALITIES
- DIFFERENCES IN MODEM MODULATION TECHNIQUES
- DIFFERENCES IN MODEM LINE TURN AROUND TIMES
- DIFFERENCES IN ERROR HANDLING AND ERROR RECOVERY
- DIFFERENCES IN DOMESTIC AND/OR INTERNATIONAL TELEPHONE NETWORKS
- DIFFERENCES IN MODEM ORDER PROCEDURES

HP 3000
MTS/3000 TERMINAL CONFIGURATIONS


HP 3000
COMPUTER
SSLC
SSLC
SSLC
RS232C UP TO 50 FEET
RS232C UP TO 50 FEET
RS232C
UP TO 50 FEET
MODEM
POINT
TO
POINT
LINE
MODEM
ONE
REMOTE
SITE
2645
2645
SYNCHRONOUS
2645
UP TO 2000'
2645
(MAY BE ASYNCHRONOUS
OR SYNCHRONOUS
HARDWIRED)
LOCAL MODEM
MODEM
LOCATION: A
REMOTE MODEM
FIRST LOCATION:A
REMOTE SITE
MODEM
REMOTE MODEM
SECOND LOCATION: B
REMOTE SITE
MODEM
MULTIDROP
LINE



MTS-TERMINALS AT LOCATION B

T_n

T_2

T_1

LOCAL MODEM LOCATION: A

REMOTE MODEM LOCATION: B

FROM SSLC BOARD

HP 3000

MODEM

LEASED LINE

MODEM

MODEM SHARING DEVICE (DIGITAL NODE) AT LOCATION B

MODEM AT LOCATION B

REMOTE MODEM LOCATION: C

MODEM

MODEM

T_1

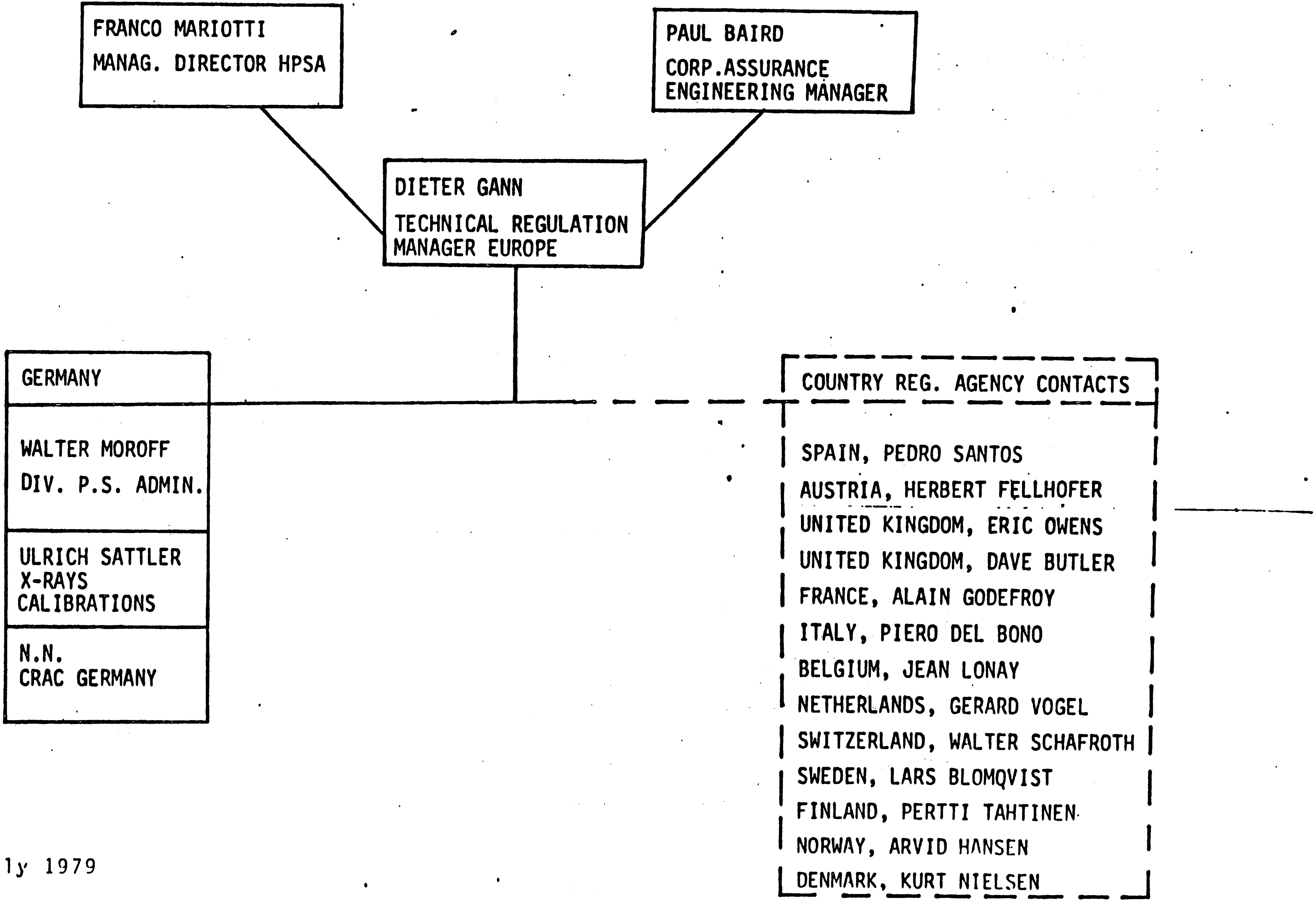
T_2

T_n

MTS - TERMINALS AT LOCATION C

IMPLEMENTATION OF A MULTI DROP CONNECTION USING A DIGITAL NODE CONFIGURATION

HPSA REGULATORY AGENCY NETWORK (CRAC)



FRANCO MARIOTTI
MANAG. DIRECTOR HPSA

PAUL BAIRD
CORP.ASSURANCE
ENGINEERING MANAGER

DIETER GANN
TECHNICAL REGULATION
MANAGER EUROPE

GERMANY

WALTER MOROFF
DIV. P.S. ADMIN.

ULRICH SATTLER
X-RAYS
CALIBRATIONS

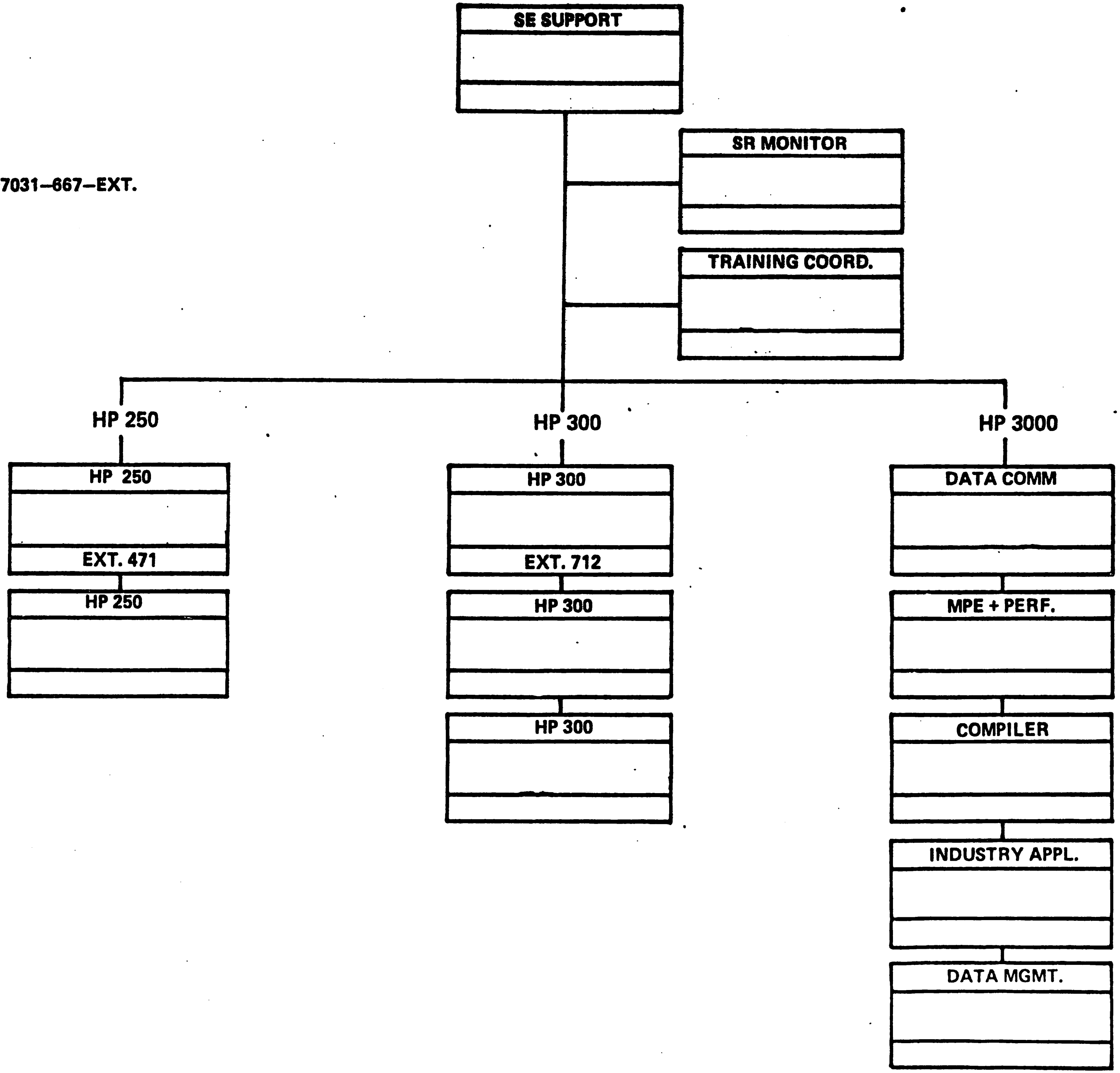
N.N.
CRAC GERMANY

COUNTRY REG. AGENCY CONTACTS

- SPAIN, PEDRO SANTOS
- AUSTRIA, HERBERT FELLHOFER
- UNITED KINGDOM, ERIC OWENS
- UNITED KINGDOM, DAVE BUTLER
- FRANCE, ALAIN GODEFROY
- ITALY, PIERO DEL BONO
- BELGIUM, JEAN LONAY
- NETHERLANDS, GERARD VOGEL
- SWITZERLAND, WALTER SCHAFROTH
- SWEDEN, LARS BLOMQVIST
- FINLAND, PERTTI TAHTINEN
- NORWAY, ARVID HANSEN
- DENMARK, KURT NIELSEN

DG/July 1979

TEL. 07031–667–EXT.


SE SUPPORT
SR MONITOR
TRAINING COORD.
HP 250
HP 300
HP 3000
HP 250
EXT. 471
HP 250
HP 300
EXT. 712
HP 300
HP 300
DATA COMM
MPE + PERF.
COMPILER
INDUSTRY APPL.
DATA MGMT.

EUROPEAN REGULATORY AGENCY NETWORK

EUROPEAN DATACOMM APPROVAL SUMMARY (MAY 79/R. DALPRA)

	V24 CONNECTION	MODEM AND MONOPOLY
AUSTRIA	NO V24 APPROVAL REQUIRED AS AT FEB 77 BUT MAY CHANGE C. 1980. PROBABLY TAKE OVER 1 YEAR. NO SPECIAL APPROVAL LABEL. NO STATUS INDIC. REQUIRED.	MODEMS TO BE PTT SUPPLIED FROM C.1980
BELGIUM	PER INSTALLATION APPROVAL IN TEMPORARY PHASE. FORMAL AGGREGATION FOLLOWS. 7-12 MONTHS; 150-750 DOLLARS. NO SAFETY APPROVAL REQUIRED. NO SPECIAL APPROVAL LABEL. NO STATUS INDIC REQUIRED.	* MONOPOLY UP TO 4800 BD 4800 BD IN TRIAL STATUS 2400 BD IN OPERATION
DENMARK	NO V24 APPROVAL REQUIRED IF CARD DECLARED V24 COMPATIBLE.	
FINLAND	V24 APPROVAL REQUIRED IF PTT MODEM USED. 1 MONTH; FREE. GERMAN APPROVAL ACCEPTED (DEE)	MONOPOLY ON DIAL-UP. MODEM TESTS INCLUDE VIBRATION TEMPERATURE AND HUMIDITY
FRANCE	NOT REQUIRED. ON SITE CHECK FOR V24 COMPLIANCE BY PTT.	
GERMANY	APPROVAL REQUIRED ON PTT LEASED OR DIAL-UP LINES. 2-6 MONTHS; 200 $ / FTZ; VDE SELF CERTIFICATION REQUIRED. APPROVAL LABEL REQUIRED. DSR (DATA SET READY) INDIC. REQUIRED FOR ATTENDED OPERATION.	MODEM MONOPOLY ON DIAL-UP AND LEASED LINES BETWEEN DIFFERENT COMPANIES

MAY '79/JUNE '79
DG/RP

EUROPEAN REGULATORY AGENCY NETWORK

EUROPEAN DATACOMM APPROVAL SUMMARY (MAY 79/R.DALPRA)

	V24 CONNECTION	MODEM AND MONOPOLY
HOLLAND	NOT REQUIRED.	NO MODEM MONOPOLY. TD 285 TECH REQUIREMENTS OUT OF DATE. NOW TD 280.0,280.5, ETC.
ITALY	NOT REQUIRED.	MODEM MONOPOLY ON DIAL-UP
NORWAY	NOT REQUIRED.	MODEM MONOPOLY
SPAIN	NOT REQUIRED.	MODEM MONOPOLY
SWEDEN	* FORMAL APPLICATION WITH PTT FORMS. SAMPLES OF FORMS HAVE BEEN SENT ON JUNE 15,1979	MODEM MONOPOLY
SWITZERLAND	NOT REQUIRED AT PRESENT IF CARD DECLARED V24 COMPATIBLE 2/77	MODEM MONOPOLY ON DIAL-UP
UK	REQUIRED WITH PTT MODEMS. 2-4 MONTHS; 400 $. NO SEPARATE SAFETY APPROVAL REQUIRED. NO SPECIAL LABEL REQUIRED. DSR (DATA SET READY) INDIC REQUIRED FOR ATTENDED OPERATION. ISOLATORS SIMPLIFY SAFETY PART OF PTT APPROVAL.	MODEM MONOPOLY ON DIAL-UP UP TO 2400 BD. THIS SPEED LIKELY TO INCREASE SOON.

MAY '79 / JUNE '79
DG/RP

INTERNATIONAL LINKS

What to expect when operating on foreign networks.
Before a user prepares for an international hookup, he must know the limits on service available and rectify some misconceptions.

For the accomplished telecommunication professional, implementing an international corporate data communications network for the first time can be a frustrating and confusing experience.
Available services, tariffs, public policy and issues affecting users can vary greatly from country to country.
And each country presents its own complexities and unique circumstances.
Six primary services available throughout the world are domestic and international telephone, domestic and international TELEX and deomestic and international private leased circuits.

Thanks to domestic and international telephone services, any nation can join the world telephone network and thus be effeciently and effectively reached by international telephone.

In the more highly developed Western European countries, leased telephone services and circuits may be readily available and rapidly installed or be subject to extensive delays based upon the existing availablility or shortage of physical facilities in each locality.

The local subscriber to telephone services in London, Paris, or Madrid for instance, may find that telephone service can be quickly installed in one part of the city, but take from six months to a year for installation in another section of the same city.

In addition, although basic telephone equipment may be available rapidly, a particular type of private branch exchange or associated telephone equipment may not be.

It is less difficult to connect two international points than to link local circuits within a country.

The company, that orders a circuit between, say, Manchester, England and Barcelona, Spain will find that the international link between the hub cities, London and Madrid, will be installed relatively quickly. The local link within the country is a different matter, however. As with telephone, local service is often subject to a variety of factors that vary from point to point. In most developed countries, a point-to-point private leased circuit can be obtained in six months or less.

There are notable exceptions and in these case circuit installation can take up to a year or more.
Users also have to contend with a limited choise of modems and equipment because of varying administration and PTT regulations.

The costs for international private leased circuits is two to lo times the cost, on a mileage basis, for similar quality circuits in NOTH AMERICA.

The cost of a quality telephone circuit between any points in the United States and Western Europe varies from a minimum of approximately $ 9.ooo to a maximum of approx. $ 15.ooo per month depending on the local termination points at each end.

It should also be noted that the terms used in defining the nature and quality of international and domestic circuits outside of NORTH AMERICA are not those with which U.S. users are familiar.

C-2-type conditioning is an unheard of acronym outsinde of NORTH AMERICA.
The more common terms include M 111, M 1o2, or M 58 as the designation of circuit quality to be installed.

Each type has its own CCITT definition.

A number of other specialized services exist in the international market place, but are only available on a limited basis.
In fact, such services are often not offered throughout an entire country.

PUBLIC NETWORKS a long way off:

A considerable number of countries in Western Europe and Asia are in the process of developing new packet-switching public data networks that will be in place and available to the enduser during the next few years.
The availability, tariff and reliability of these networks are yet to be established, and various countries are still concentrating on such fundamentals as common protocols and interface specifications. So the use of these new networks cannot yet be relied upon by the large user in establishing short-term plans for international data communications.

Regulations and policies differ from country to country within the same region and even for countries in similar economic states. Also, users in most every foreign nation have little resource to appeal, alter or change a policy with which they are unhappy. In more developed countries, such as England, France, West Germany and Japan, regulation and policy are formal activities whereby general official guidelines or documents exist that provide the user with a summary of what he can and cannot do. New regulations are developed after careful consideration

by a formal committee or group and discussion with users.
In many countries, regulation and policy evolve on a de facto basis and are subject to revision or change without any formal procedure. Often, policy is changed as the result of an internal meeting or discussion between two or more individuals within a particular administration.

There are 18 CCITT study groups and approximately a dozen other specialized organizations and regional planning groups that are concerned with literally hundrets of standards and regulatory issues.

PTT policies and issues that effect users maintaining international private networks are:

- TIMESHARING AND DATABASE SERVICES
- JOINT USE AND LINE SHARING
- DATA ACROSS BORDERS
- DATA TRANSMISSION ON THE TELEPHONE NETWORK
- AUTODIAL TELEX
- ENCRYPTION
- NETWORK PROTECTION **
- COMMON MISCONCEPTIONS

** NETWORK PROTECTION

Almost every administration on PTT requires that a new type of device be submitted for approval before being installed by a user or on a public facilitiy.
In theory, the reason for requiring approval is to protect a PTT's network and facility from damage that can be caused by non-standard or faculty equipment.

In addition it is mandatory in some countries, that users obtain modems and terminals for some or all services directly from the PTT!

Some PTTs require the user to provide his own equipment or terminals from an established list of type-approval devices, while others optionally provide equipment or allow the user to select his own from an approved list.
In all cases, the user cannot select a device unless it is either provided or approved in advance by the PTT.

A recent complication in type approvals is the tendency of modem manufacturers to integrate modems or line separator devices into a more sophisticated terminal, multiplexer, or other equipment. When this occurs, it is difficult to tell whether the individual subassembly or the entire integrated device must be type approved.

Various PTTs have adapted different requirements concerning this issue.

TOOLS AND TECHNIQUES AVAILABLE FOR PERFORMANCE IMPROVEMENT

HJ 4

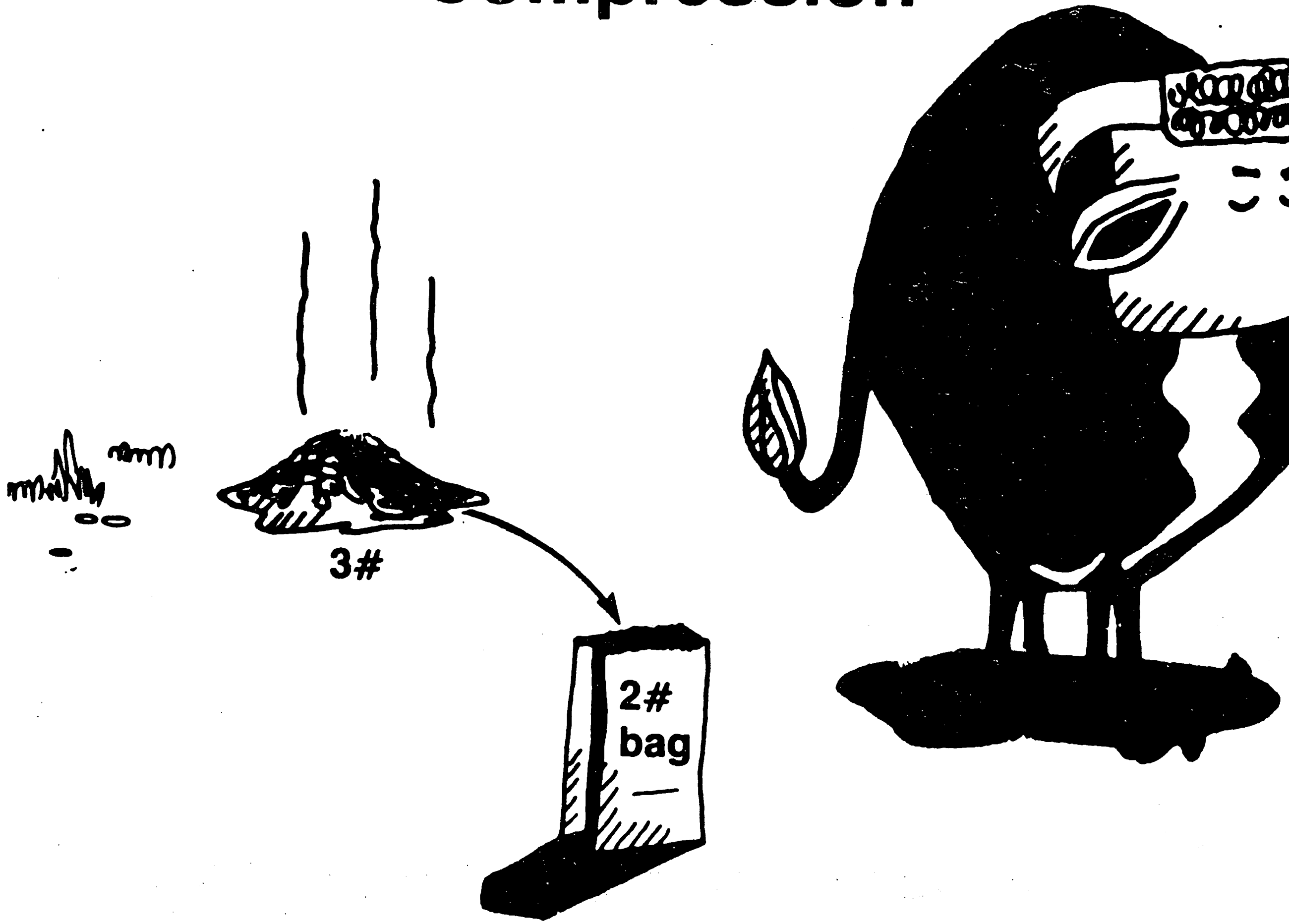
To evaluate a system,the user need a set of performance criteria that encompasses both the carrier facilities and the data communications hardware and treats them as a single system. There are nine criteria I can think off for accessing how well a system handles information interchange from the user's viewpoint.

It also presents typical values to apply as yard sticks in comparing services and equipment.

The nine parameters do not represent all possible performance criteria but are the most essential factors.
The need for such criteria has become all the more urgent with the introduction of such alternatives to the voice-grade telephone system as packet switching, satellite links, and all-digital terrestrial networks. This development has complicated system evaluation because the parameters of these carriers are sometimes made under different conditions or differently defined from those of the conventional voice-grade telephone lines. But the criteria mentioned here can be applied to any system, whatever the natur of the carrier and porvide the user with a much-needed basis for comparison of dissimilar systems.

- TRANSFER RATE
- AVAILABILITY
- RELIABILITY
- ACCURACY
- CHANNEL-ESTABLISHMENT TIME
- NETWORK DELAY
- LINE TURNAROUND DELAY
- TRANSPARENCY
- SYSTEM SECURITY

Compression



How can you put 3#'s into a 2# bag?

Compress it!

Performance Optimization

Data transfer

Time & system load is function of the number of DS requests

1) Improve with:
   a) Use blocked file transfer with file created with large blocking factor
   b) Use PTOP and pack buffer
   c) Prepare special file for transfer with larger record size
2) Use largest line buffer size commensurate with error rate
3) Use compression where appropriate

Performance Optimization (cont.)

Interactive activity

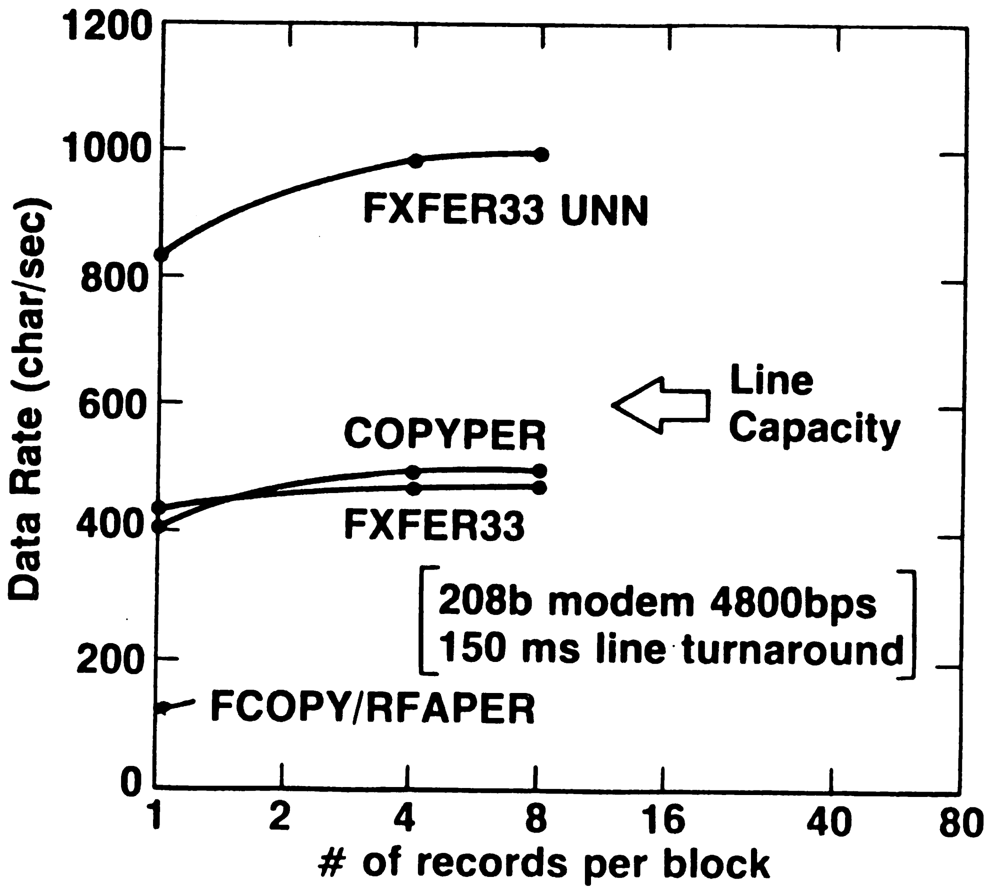
Execute repetitive operations on appropriate system using stream jobs or UDC's.

Data base activity

Program so that all data base access is local. Use PtoP programs to transfer summary information (80/20 rule)

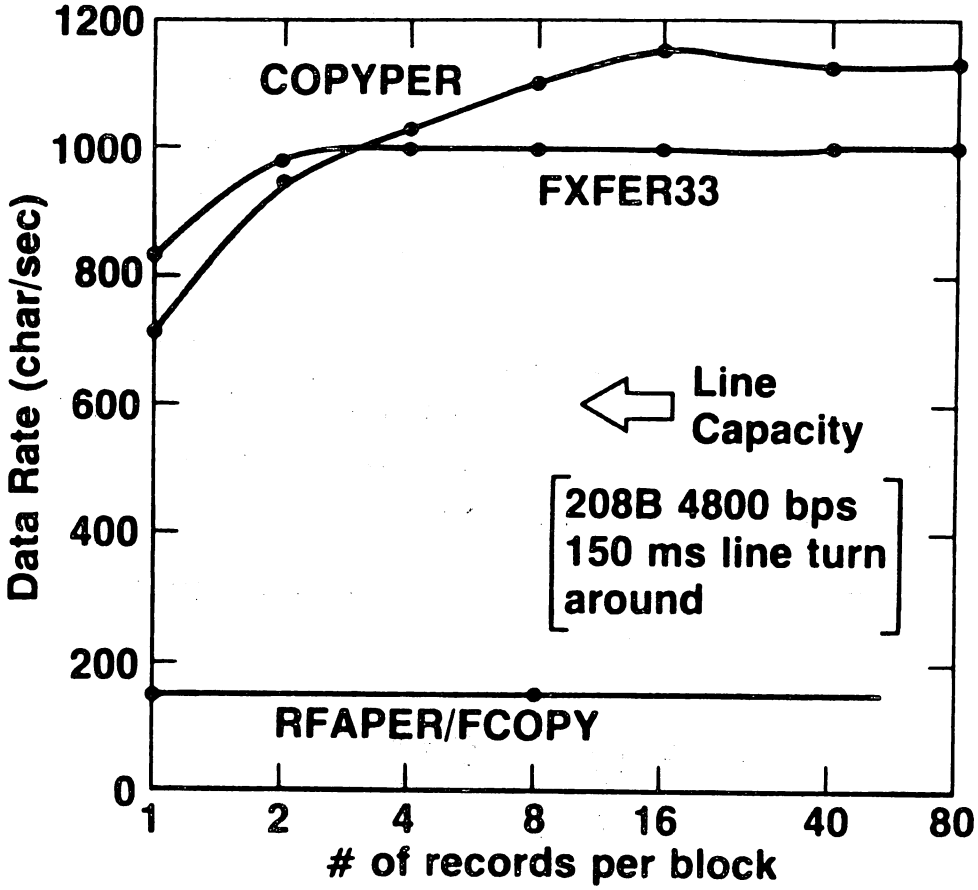
Verify operation as designed with CSTRACE.

SSLC (nocomp)
Data Rate

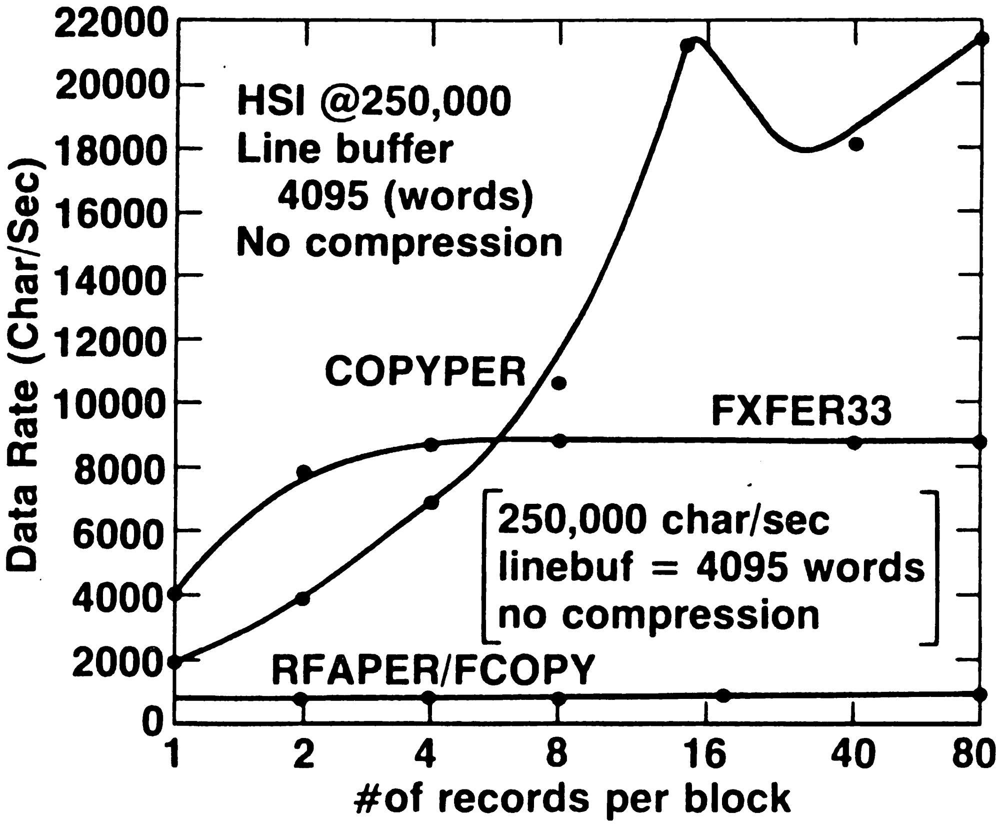


208b modem 4800bps
150 ms line turnaround

SSLC (comp) Data Rate



HSI
Data Rate



DATA COMMUNICATIONS STANDARDS

Effective communication of data among computer centers and remote terminals is essential to maximum utilization of current data processing resources.
However, there are significant compatibility problems to be solved due to the broad differences inherent in existing ADP and communications systems an equipment.

Steps are being taken to solve these problems by identifying and developing uniform standards and practices to satisfy the needs of specific, and in some cases general, communities of interest.

A number of organizations, including professional societies, trade associations, government agencies at both the Federal and state levels and national and international standards bodies, are invlolved in these activities.

A review of the working relationships among these organizations, their current efforts, and some 50 existing standards per training to data communications should be of interest to everyone concerned with information processing and communications.

To an increasing extent telecommunication networks, data links, and communication terminal facilities are employing computers.
There are presently in excess a broad number of computers installed worldwide and not all of these existing computers are involved in teleprocessing operations.

Such computer applications, involving communication networks and facilities, may be classified according to one of two general categories:

- Those computer applications that are concerned with the use of computers as a part of the communications function, for network control, circuit or message switching, or customer billing operations.

- Those computer applications that employ data communications facilities and networks to provide information interchange to remote locations.

The first application category is the principal concern of the communication system specialist in that computers are employed primarily in support of the communication function.

The second category, frequently referred to as teleprocessing, is primarily the responsibility of the information processing system designer and ADP specialist.

Here, communication facilities are used to support the computer users requirements for the remote processing and interchange of information.

Personnel in either application category generally require varying degrees of interdisciplinary knowledge of both technologies - computer and communications.
This is particulary true in the areas of system design, operating procedures and standards.

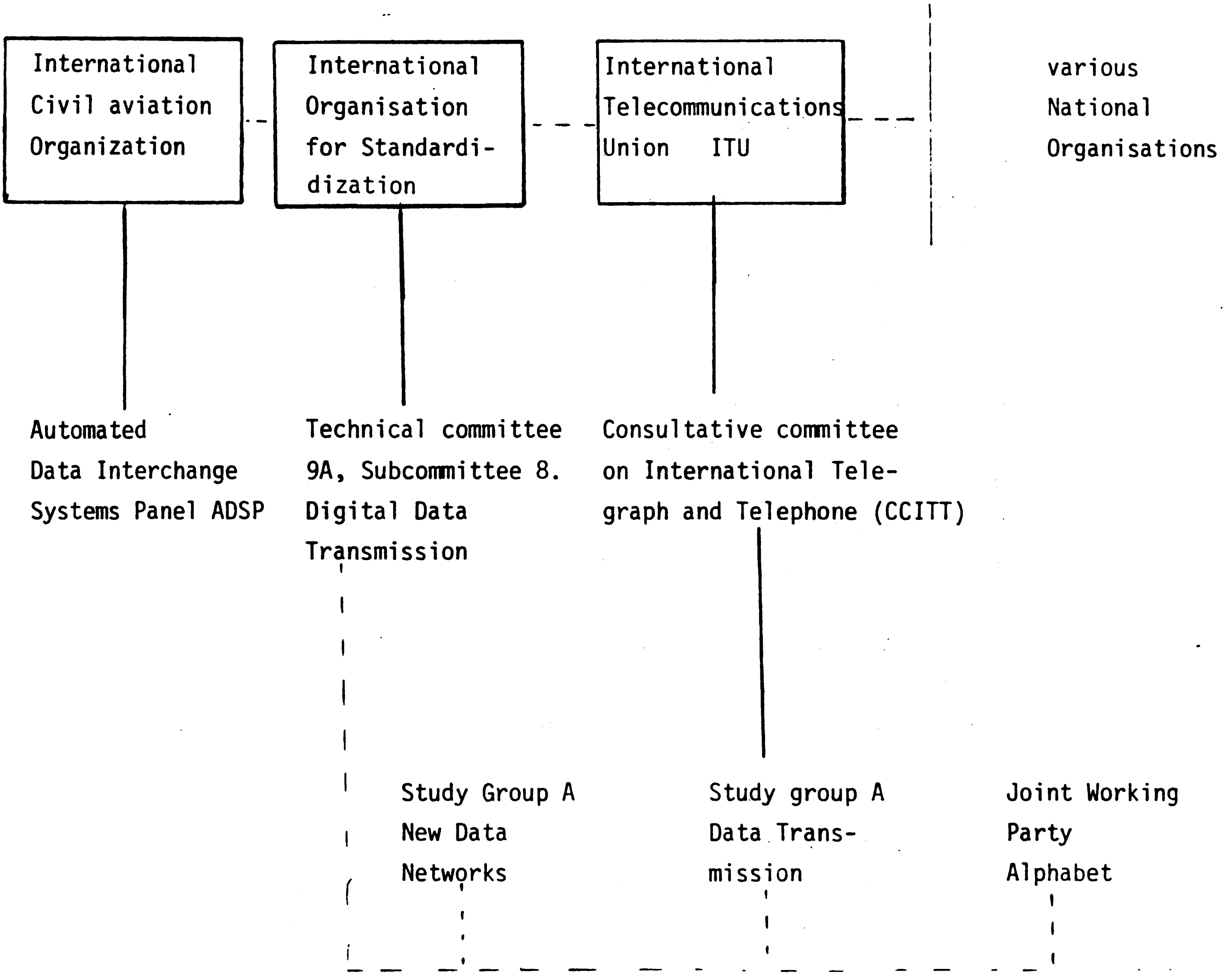
The organisations that deal with standards development for telecommunications technology are both older and separate from those corresponding organizations established for information processing standardization. Furthermore, because of fundamental differences among the industries servicing these technologies as well as among the population of users there are significant differences in both the standards development and implementation processes.

However, the increasing interaction between computer and communication systems has created a growing awareness of the need of consistent procedures and common practices between these two technologies, particularly with respect to data communication standards.

As a result, both technologies have established formal standards coordination and review arrangements among their respective trade associations, industry groups and national and international organization dealing with data communication standards.

Various internal coordination arrangements have also been evolved by indivuduals who cross the technological boundary to participate in both communications and computer standards activities.

Organizations concerned with data communications standardization and some of their more formal coordinating arrangements are outlined in Figure 2:


International Civil aviation Organization
International Organisation for Standardi-dization
International Telecommunications Union ITU
various National Organisations
Automated Data Interchange Systems Panel ADSP
Technical committee 9A, Subcommittee 8. Digital Data Transmission
Consultative committee on International Tele-graph and Telephone (CCITT)
Study Group A New Data Networks
Study group A Data Trans-mission
Joint Working Party Alphabet

ISO / TC 97 / SC-6:

Internationally, there are a number of organizations involved either directly or indirectly with the development of data communication standards.
FOREMOST among these is the INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), Technical Committee (TC) 97, Subcommittee (SC) 6 concerned with digital data transmission.

Membership in this subcommittee, as in all ISO activities, is provided by the national standards organizations of the participating countries.
The scope and program of work of this group include the continual surveillance of the telecommunication environment as it affects the development of data communications standards and the ability to build, install, operate and maintain data processing systems using telecommunications.
This surveillance involves monitoring the technologies employed by the member country's common carrier and postal telephone and telegraph services, so as to take into account the communication environment in developing standards and to react when these services do not meet the requirements of the data processing community.

ITU: The International Telecommunications Union is an intergovernmental organization concerned with establishing international agreements, treaties and standards with respect to telecommunication.
It seeks to coordinate the works of national governments and private bodies by promoting the development of technical facilities and their efficient operation and to improve the efficiency, usefulness, and availability of telecommunication services.
The ITU studies technical, operating and tariff questions relating to the telecommunications, makes recommendations, and collects, publishes and disseminates information for its memebership.

CCITT (ITU): The Consultative Committee on International Telegraph and Telephone is a technical group operating under the ITU established to examine and make recommendations on technical, operational, and tariff matters relating to telegraphy, facsimile, and telephony.CCITT membership is made up to five categories.

- member country telecommunication adminstrations (A-members);
- recognized private agencies as AT & T (B-members);
- scientific or industrial organizations such as equipment manufacturer associations (C-members);
- international organizations such as the INTERNATIONAL OIVIL AVIATION ORGANIZATION and the ISO (D-members)
- specialized agencies operating in related fields such as the WORLD METEOROLOGICAL ORGANIZATION (E-members)

Only A-type members have voting powers at the plenary session where decisions are made.
e.g. The United States is represented on CCITT by the Department of State with domestic common carriers, the FCC and the Office of Telecommunications Policy serving as its principal technical consulting agents.
Data Communication standards are coordinated in CCITT affairs through direct participation of C and D type members that include ISO and the European Computer Manufacturers Associations (ECMA).

The CCITT program of work is divided over some 25 STUDY GROUPS , WORKING PARTIES AND AREA plan committees who discuss and prepare the detailed recommendations for their particular areas of specialty.

Three groups are of prime interest to the data communications community:

- Special Study group A.
  Data Transmission, is the principal point for the data processing community's contact with CCITT. The work progress includes all matter relevant to data transmission including interfaces, modems and maintenance procedures.

- JOINT WORKING PARTY ALP,
  Alphabet is responsible for work on alphabets, codes and related matters pertaining to character transmission structure and keyboards.

- STUDY GROUP VII, NEW DATA NETWORKS
  is responsible for working on new data networks, with the emphasis on digital (in contrast to a alog) networks for data transmission and related uses.

ADISP (ICAO) the International Civil Aviation Organization is one of a number of intergovernmental application oriented organizations whose constituents are involved with international commerce and who are increasinlgy dependent on data interchange in the conduct of their respective business.

ICAO has established the Automated Data Interchange System Panel (ADISP) to develop data communication standards to be employed in international air traffic control.

UNRESOLVED ISSUES IN DATA COMMUNICATION STANDARDS

As of this writing, a number of outstanding issues in the data communication standards arena remain to be resolved. Each of the previously described standards organizations is involved to varying degrees in attempting to resolve these issues.

Perhaps the most controversial of these issues is related to bit-oriented control procedures for terminal-to-terminal and terminal-to computer data interchange.

Some domestic and international groups are anxious to reach agreement on point-to-point, two-way simultaneous modes of operation, leaving the development of standards for multi-point for a later date. Others are just as adamant that only a single standard be developed covering the many different modes of operation that are possible.

The resultant differences of opinion probably will delay any agreements on an initial standard by as much as 2 years.

Another outstanding issue is the technique to be standardized for providing the necessary controls for point-to-point, two-way simultaneous operation.
Substantial agreements had been reached as recently as 1971 on the subject and some standards were written into the Annex to the ICAO contract based on this concept of data link control procedures.

However, due to the desire on the part of some international bodies to develop a multi-point procedure which could be incorporated in a single standard, and the delays that ensured, another concept for point-to-point two-way simultaneous operation has been introduced. This issue has somewhat simplistically been called the single numbering controversy.

At this moment, still much need to be done to resolve the differences that have cropped up on this subject.

A number of interesting developments have occured within the last year in the area of data communications standards. Some task groups have developed a draft standard on "MESSAGE HEADING FORMATS" for Information Interchange Using the ASCII for Data Communications System Control. This is a character-oriented network control system.

The same document or its equivalent, was introduced into the international area at the ISO/TC-97/SC-6 Meeting in Stockholm 1973. However, a number of international organizations, such as ICAO and IATA are working on bit-oriented network control procedures. There is some question as to the advisability of introducing a character-oriented control procedure for networks since it appears that most of the newer networks being planned will utilize the bit-oriented rather than the character-oriented procedure.

A great deal of attention is now being given to the relationship between the value added networks, such as the ARPA Network, the SITA Network, the CIDIN etc. and the public data networks now being planned by the BELL SYSTEM and other international organizations in the United Kingdom, Germany and France.

It is increasingly apparent that the interworking of these very closely related networks will not be possible unless close attention is given to the standards that are developed for these systems.

Internationally, ISO/TC-97/SC-6 has initiated projects in the public data network area and has formed task groups which will work closely with the working groups on control procedures. CCITT, COMMITTEE SPECIAL A, has established a project on packed switching which is one of the "VALUE ADDED NETWORKS" (VAN) activities.

The various groups working on these closely connected projects are analyzing the existing standardization process in light of these networks.

They have been considering the restructuring of their organizations to meet the challenges of this new technology for which very little, if any precedent has been established.

EXISTING DATA COMMUNICATION STANDARDS AND RECOMMENDATIONS

The approved international recommendations of ISO and CCITT plus several others (eg. ANSI, EIA etc.) relating to data communications are provided below:

INTERNATIONAL RECOMMENDATIONS (ISO)

ISO	1155 ... 1969	The use of Longitudinal Parity to detect errors in Information Messages.
ISO	1177 ... 1970	Character Structure for START/STOP and Synchronous Transmission
R	1745 ... 1971	Basic Mode Control Procedures for Data Communication Systems.
ISO	211o ... 1972	Data Communication - Data Terminal and Data Communication Equipment Interchange Circuits Assignment of Connector Pins.
ISO	2111 ... 1972	Data Communication-Basic Mode Control Procedures-Code-Independent Information Transfer.
ISO	2593 ... 1973	Connector-Pin allocations for the High Speed Data Terminal Equipment.
ISO	2628 ... 1973	Basic Mode Control Procedures - Complements
ISO	2629 ... 1973	Basic Mode Control Procedures - Conversational Information Message Transfer

INTERNATIONAL RECOMMENDATIONS (CCITT)

V. 1.	Equivalence between binary notation symbols and the significant conditions of a two-condition code.
V. 2.	Power levels for data transmission over telephone lines
V. 3.	International alphabet number 5 for transmission of data and messages.
V. 4.	General structure of signals of the 7-unit code for data and message transmission.
V.10.	Electrical characteristics for unbalanced double-current interchange circuits
V.11.	Electrical characteristics for balanced double-current interchange circuits
V.13.	Answer back unit similators.
V.15.	Use of acoustic couplers for data transmission
V.21.	200-Baud Modem standardized for use in the general switched telephone network
V.22.	Standardization of modulation rates and data-signalling rates for synchronous data transmission in the general switched telephone network
V.22B.	Standardization of modulation rates and data-signalling rates on leased telephone circuits.
V.23.	600/1200 Baud Modem standardized for use in the general switched telphone network

V. 24.	Functions and electrical characteristics of circuits at the interface between DATA TERMINAL EQUIPMENT and DATA COMMUNICATION EQUIPMENT.
V. 25	Automatic calling and/or answering on the general switched telephone network.
V. 26	2400 Bits/s modem for use on four-wire leased point-to-point circuit
V. 26B	2400 Bits/s modem for use on the general switched telephone network.
V. 27	Modem for data signalling rates up to 4800 Bits/sec. over leased circuits
V. 28	Electrical characteristics for interface circuits
V. 30	Parallel data transmission systems for universal use on the general switched telephone network
V. 31	Electrical characteristics for contact closure type interface circuits.
V. 35	Transmission of 58 Kilobits/second data using 60 to 108 KHz group bank circuits
V, 40	ERROR indication with electro-mechanical equipment
V. 41	Code-independent error control system
V. 50	Standard limits for transmission quality of data tranmission
V. 51	Organization of the maintenance of international telephone - type circuits used for data transmission.

V. 52	Characteristics of distortion and error rate measuring apparatus for data transmission
V. 53	Limits for the maintenance of telephone type circuits used for data transmission
V. 56	Comprehensive tests for modems that use their own interface circuits
V. 57	Comprehensive test set for high transmission rates

X. 1.	User classes of service and data signalling rates for public data networks
X. 2.	Recommended user facilities available in public data networks
X. 20.	Interface between data terminal equipment and data circuit terminating equipment for start-stop services in use classes 1 and 2 on public data networks.
X. 21.	Interface between data terminal equipment and data circuit terminating equipment for synchronous operation on public data networks.
X. 50.	Fundamental parameters of a multplexing scheme for the international interface between synchronous data networks.
x. 70.	Terminal and transit control signalling system for start-stop services on international circuits between anisochronous data networks.

AMERICAN NATIONAL STANDARDS (ANSI)

X. 3.1. ...	1969	Signalling Speeds for Data Transmission
X. 3.4.	1968	American Standard Code for Information Interchange
X. 3.15 ...	1966	Bit Sequencing of the U.S.A. Standard Code for Information Interchange in Serial-by-Bit Data Transmission.
X. 3.24	1968	Signal Quality at Interface between Data Processing Terminal Equipment and Synchronous Data Communication Equipment for Serial-Data Transmission (EIA RS 334)
X. 3.25	1968	Character Structure and character Parity Sense for Parallel-by-Bit Data Communication in the U.S.A. Standard Code for Information Interchange.
X. 3.28. ...	1971	Procedures for the use of the Communication Control characters f the U.S.A. Standard Code for Information Interchange in Specified Data Communication LINKS.

ELECTRONIC INDUSTRY ASSOCIATION STANDARDS (EIA)

RS 232 C	Interface between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange
RS 334	Signal Quality at Interface between Data Terminal Equipment and Synchronous Data Communication Equipment for Serial Data Transmission.
RS 366	Interface between Data Terminal Equipment and Automatic Calling Equipment for Dat Communication.

" HP Software Development in Manufacturing "

Rich Zalisk

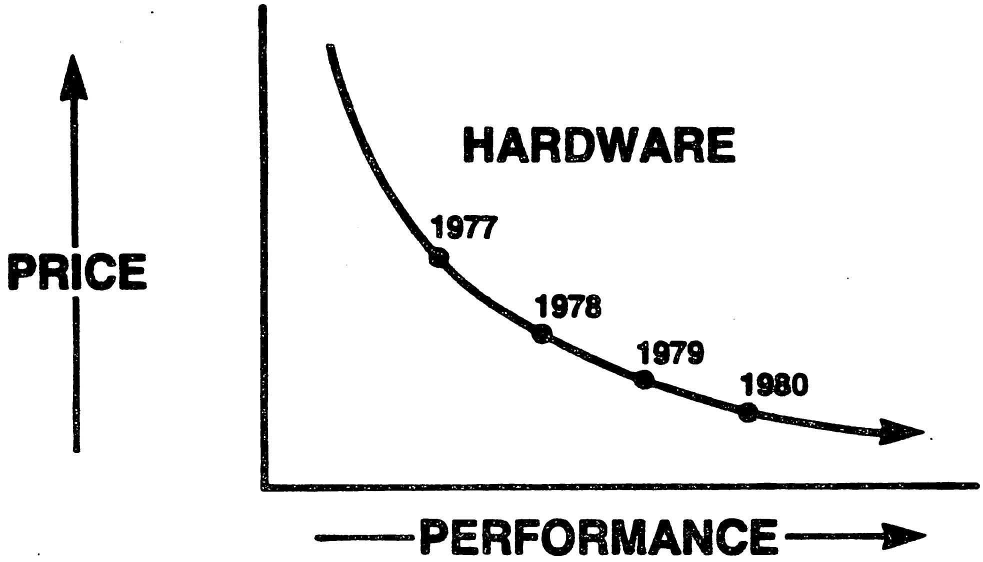
John Stark

HP MANUFACTURING APPLICATIONS

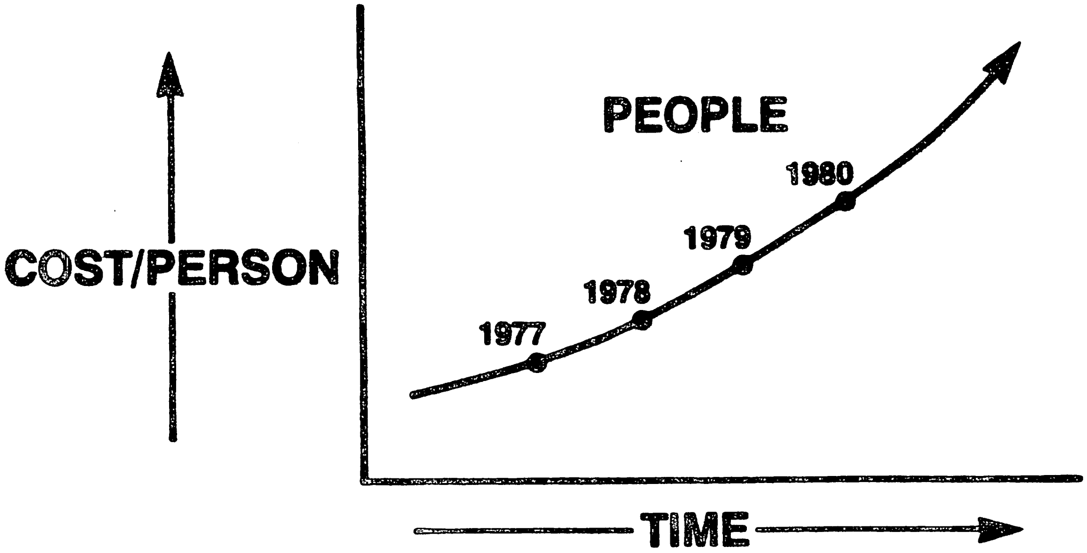
AGENDA

- BACKGROUND AND TRENDS
- STRATEGY
- MATERIALS MANAGEMENT: MFG/3000
- QUANTITATIVE MODELS
- QUESTIONS

FAMILIAR TRENDS



AND



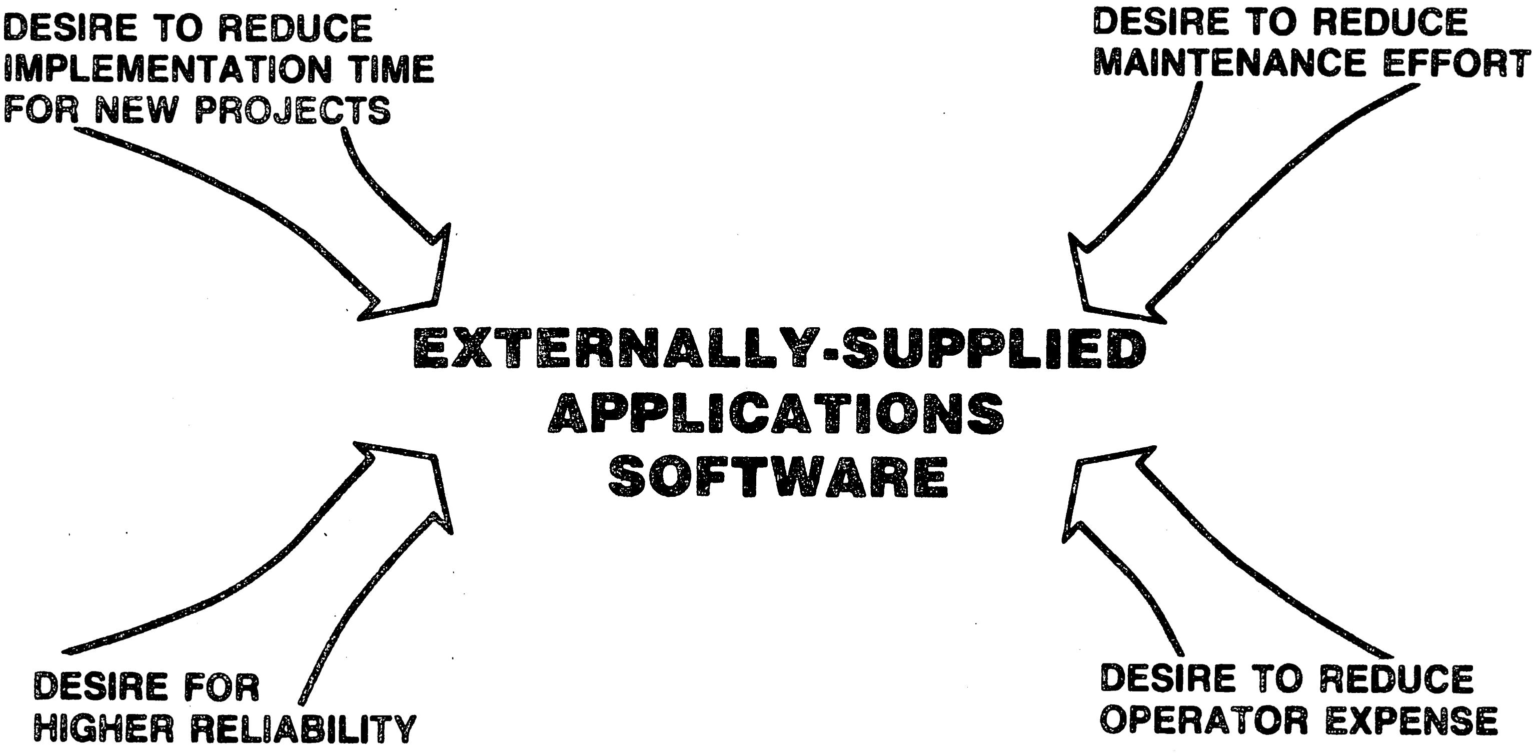
PROFILE OF EDP BUDGET

	PROFESSIONAL PEOPLE			HARDWARE	TOTAL
	SYSTEM ANALYSTS	PROGRAMMERS	OPERATORS		
NEW PROJECTS	9%	4%	—	2½%	15½%
MAINTENANCE	11%	12%	½%	5 %	28½%
OPERATIONS	—	4%	9½%	42½%	56 %
TOTAL	20%	20%	10%	50%	100%

CONCLUSIONS:

1) CONSTANT PRESSURE TO REDUCE OPERATIONS COSTS
2) NEVER ANY RESOURCES AVAILABLE FOR NEW PROJECTS
3) MAINTENANCE IS AN UNDESIRABLE "PAIN IN THE NECK"

TRENDS



DESIRE TO REDUCE IMPLEMENTATION TIME FOR NEW PROJECTS

DESIRE TO REDUCE MAINTENANCE EFFORT

EXTERNALLY-SUPPLIED APPLICATIONS SOFTWARE

DESIRE FOR HIGHER RELIABILITY

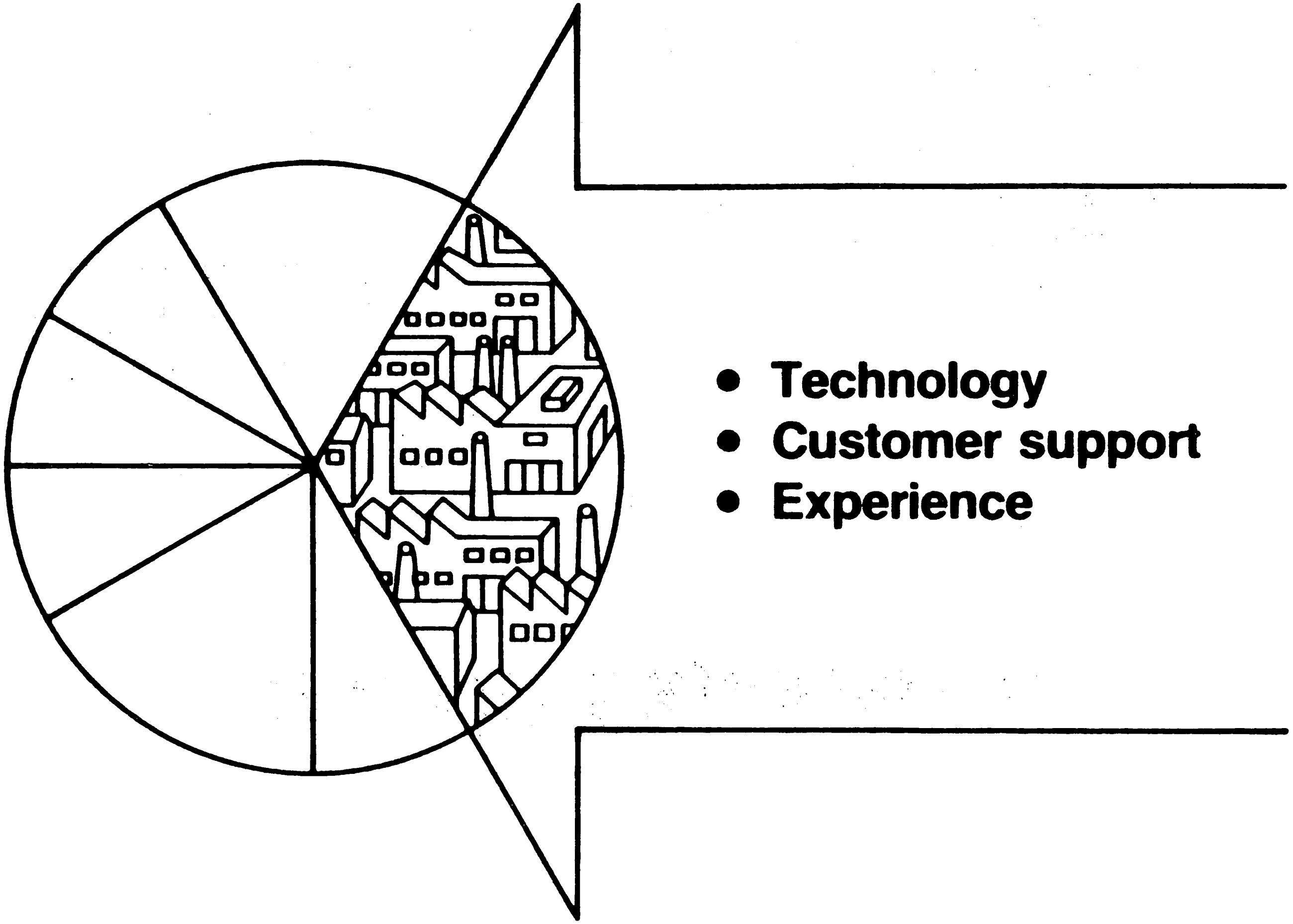
DESIRE TO REDUCE OPERATOR EXPENSE

POTENTIAL BENEFITS OF "EXTERNALLY-SUPPLIED" SOFTWARE

- NO NEW PRODUCT DEVELOPMENT CYCLE
- LOWER INTERNAL RESOURCES
- SHORTER IMPLEMENTATION TIME
- HIGHER RELIABILITY
- PREDICTABLE PERFORMANCE
- LOW PRICE
- NO MAINTENANCE/SUPPORT BURDEN
- BETTER STANDARD DOCUMENTATION
- BETTER STANDARD TRAINING
- SHARED EXPERIENCES/PROVEN SUCCESS
- PERFORMANCE OPTIMIZATION (FIRMWARE)

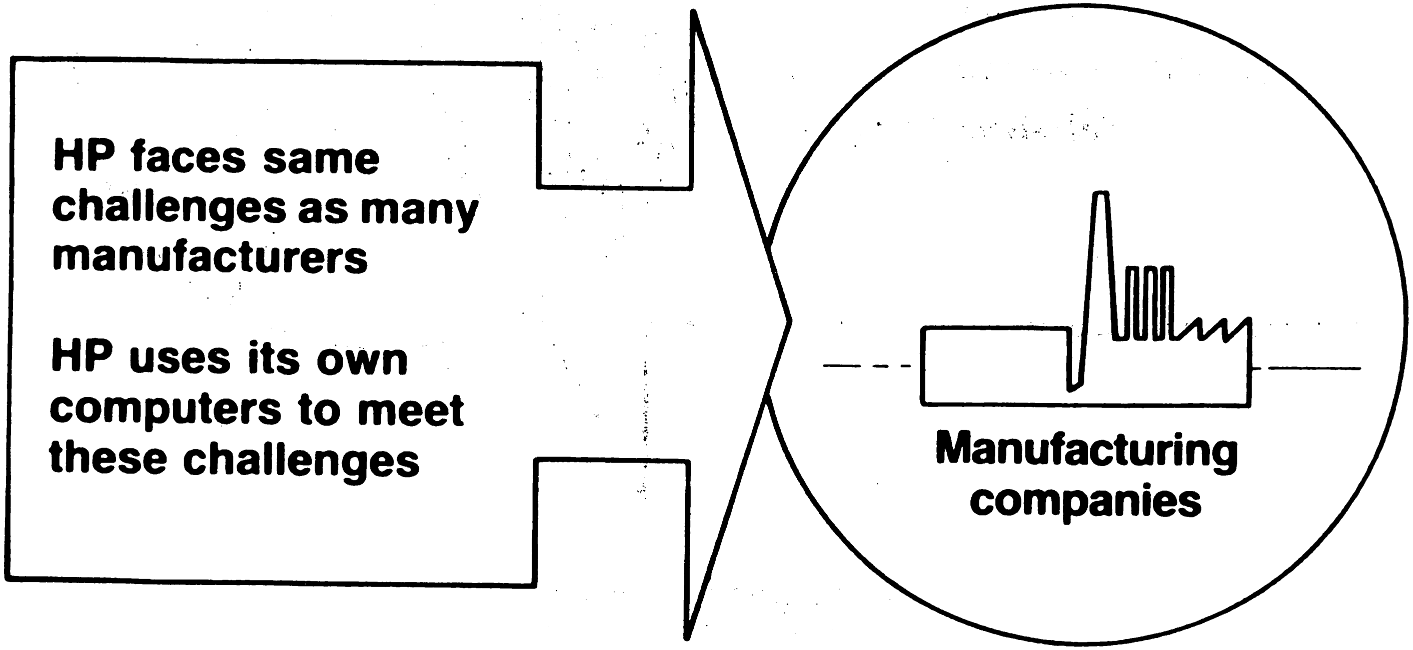
LET CUSTOMERS CONCENTRATE THEIR RESOURCES ON PROBLEMS UNIQUE TO THEIR OWN BUSINESS

Contributions to Manufacturing Systems



- Technology
- Customer support
- Experience

The HP Experience



HP faces same challenges as many manufacturers

HP uses its own computers to meet these challenges

Manufacturing companies

HP Uses HP Computers

- Material Requirements Planning
- Capacity Requirements Planning
- Sales order entry
- Process control
- Shop floor control
- Accounting

Over 350 HP Computers Solving HP's Manufacturing Problems

MFG/3000 is Materials Planning and Control Products for:

- Engineering data control
- Inventory and order status
- Materials requirements planning

MATERIAL
CONTROL

HEWLETT **hp** *PACKARD*

NEELY SALES REGION • [illegible]

FROM: John Stark

DATE: July 11, 1979

TO:

SUBJECT: Outline of Proposed Presentation for HP3000 International Users Group Meeting in Lyon, France

- I. Topic: Interfacing Quantitative Modeling Techniques to MFG3000
- II. Background
  - A. MFG3000
    - 1. What does it do
    - 2. What doesn't it do
  - B. Some useful mathematical programming models
    - 1. Exponential Smoothing
      - a) What is it
    - 2. Net Present Value
      - a) What is it
    - 3. Linear Programming
      - a) Single period
        - 1) What is it
        - 2) What are its limitations
      - b) Multi period
        - 1) What is it
        - 2) What are the advantages over single period
- III. Application of Models and MFG3000
  - A. Exponential Smoothing
    - 1. Forecast Demand
      - a) Establishes a master schedule of end items
  - B. Multi Period Linear Programming/Net Present Value
    - 1. Determines if forecasted demand will maximize net present value of return on investment to firm
      - a) Adjust master schedule to maximize net present value
        - 1) Realize that some opportunities cannot be carried over to subsequent periods
          - a.) Implies a sub-optimum solution may be necessary

C. MRP3000

1. Determines lower level requirements to build master scheduled items

D. Multi Period Linear Programming

1. Utilize to optimize mix of detail parts to be fabricated

a) Capacity constraints

IV. Example

A. Happy Peddler Bicycle Company

Interfacing Quantitative Modeling Techniques to MFG 3000

MFG 3000 is set of 3000 based application software available from Hewlett-Packard to aid in the planning for and control of raw materials. MFG/3000 consists of four modules which can be purchased separately or as an integrated package. The modules included in the current release of the software are Engineering Data Control (EDC), Inventory and Order Status (IOS), Materials Requirements Planning (MRP) and Standard Product Costing (SPC). Basically, EDC is used to structure a Bill of Materials (i.e., outline what and how many parts go into the fabrication of a final assembly item). IOS keeps track of which of these detail parts are on order (whether purchased or fabricated in-house) and how many of each detail part is available in inventory for issue. MRP looks at a company's master schedule (the schedule of end items that the company is planning to build); looks at EDC to determine all of the component parts and how many of each are required to build the end item; looks at IOS to determine if there are adequate detail parts in inventory or on order to fill the requirements generated by MRP; and finally, displays a list of suggested new orders (net of inventory) for these component parts.

Purchase orders are released for those parts that are purchased from outside vendors and production control releases orders for those detail parts that are fabricated in-house. At the appropriate time, the completed detail parts are released to the shop and the finished product is assembled. Sounds great! What else could a company possibly require to make this process work?

First, we assumed the Master Schedule existed and will always exists. Not a valid assumption! Someone has to create the Master Schedule. This is not a problem if items are put into the schedule only when an actual sales order is received. But what if we wanted to smooth out our production and try to predict over some planning horizon what the demand will be for our products? What tools does Hewlett-Packard offer in the area of forecasting?

Currently, nothing. Quantitative methods however, provides a model for forecasting demand based on past experience, but weighted to reflect any additional information which we might have. (i.e., sudden increase in sales due to some technological edge that we have over our competitors,or there are seasonal variations in our sales, etc). This technique is called Exponential Smoothing.

Once we have done a reasonable job of forecasting demand, the next question to ask is "is it profitable to make these products?" Again, Hewlett-Packard currently does not address this area with a purchasable software product, but there are models available which will aid us in making a decision. One such tool is a Multi-period Linear Programming model which utilizes discounted cash flows as inputs. Multi-period is used because we are assuming a planning horizon of more than one year and discounted cash flows are used because we are assuming that the firm wishes to maximize the net present value of its investments over time. (Building a finished good is an investment in labor, material and overhead).

Finally, we must consider the capacities of the firm in being able to actually produce the goods that we have determined will maximize the net profit of the company. Here, as before, we can utilize Multi-period Linear Programming with capacities as inputs to the model.

In summary, there are several modeling techniques available from Quantitative methods which will aid a company in better controlling its day to day operations and thereby increase its profitability. Even though Hewlett-Packard intends to address all areas of manufacturing from materials management to financial accounting, there is no need to wait for the one vendor solution. Utilize what has been available for years...Quantitative methods!

CHOICE OF SOFTWARE LANGUAGE

ERIC GARRIGUE VESELY
Vesely & Associates
Principal

Hundreds of computer languages and dialects are available to users for programming their application. A good choice saves money, time, and frustration; a bad choice can mean the complete failure of the computer installation.

The major languages include ALGOL, APL, BASIC, COBOL, FORTRAN, P/L I, RPG, and each computer's assembly language.

Each language has strengths and weaknesses. These must be carefully analyzed in the full spectrum of program design through program maintenance to arrive at the total cost of using that language. The language costing the least is the best choice.

Analysis criteria includes computer configuration, application, portability, conversion, compiler speed and aids, debugging tools, execution speed, self-documentation features, maintainability, programmer availability, other languages and software interfaces.

The following text provides tabular evaluation techniques and explanations.

CHOIX DU LANGAGE DE SOFTWARE

Des centaines de langages de calculateur sont à la disposition des utilisateurs pour programmer leur application. Un bon choix économise de l'argent, du temps, et de la frustration. Un mauvais choix peut entraîner l'échec complet de l'installation de calculateur.

Les langages les plus importants comprennet ALGOL, APL, BASIC, P/L I, RPG et le langage d'assemblage de chaque ordinateur.

Chaque langage a ses points forts et ses points faibles. Ceux-ci doivent être examinés soigneusement en considérant toutes les phases de programmation depuis le projet du programme jusqu'au programme de maintien pour déterminer le coût total de l'utilisation de ce langage. Le meilleur choix est le langage le moins coûteux.

Les critères de l'examen comprennent la configuration, l'application, la mobilité, la conversion, la vitesse du compilateur et de ses aides, les outils de mise en point, la vitesse d'exécution, les dispositifs d'auto-documentation, la maintenance, la disponibilité du programmeur, d'autres langages et les interfaces du software.

Le texte suivant présente sous forme de tableau les explications et les techniques d'évaluation.

AUSWAHL EINER SOFTWARE-SPRACHE

Es gibt hunderte von Computersprachen und -dialekte, die die Benützer zum Programmieren ihrer Anwendung gebrauchen können. Eine gute Wahl spart Geld, Zeit und Enttäuschungen; eine schlechte Wahl kann den vollkommenen Ausfall des Computers herbeiführen.

Unter den wichtigsten Sprachen sind ALGOL APL, BASIC, COBOL, FORTRAN, P/L I, RPG und die Montagesprache jedes Computers.

Jede Sprache hat ihre Stärken und Schwächen. Diese müssen genau untersucht werden, und zwar vom Gesichtspunkt des Programmentwurfes bis zur Programminstandhaltung um die Totalkosten der Anwendung der Sprache zu erhalten. Die Sprache , die am wenigsten kostet, ist die beste Wahl.

Zu den Untersuchungskriteria gehören die Computerform, die Anwendung, Beweglichkeit, Umwandlung, Compiliergeschwindigkeit und Hilfsmittel, Fehlerbeseitigungswerkzeuge, Ausführungsgeschwindigkeit, Selbstdokumentationseigenschaften, Instandhaltbarkeit, Programmierer-Verfügbarkeit, andere Sprachen und Software Übergangsflächen.

Der nachstehende Text beinhaltet tabellarisch dargestellte Berechnungstechniken und Erklärungen.

1. COMPUTER CONFIGURATION

The first choice consideration is the host computer configuration. Obviously, if a language is not available on your selected computer then it is eliminated. The tabular form is:

Computer	Cobol[1.1]	SPL[1.2]
Configuration Cost[1.3]		
Compiler Price[1.4]		
Annual Maintenance[1.5]		
Price X ____ years[1.6]		
Total Cost		
Est. No. of Compiles[1.7]		
Cost per Compile		

1.1. The languages are selected for illustration; they could easily be APL, BASIC, FORTRAN, SPL, etc.

1.2. Hewlett-Packard 3000 assembler

1.3. The cost of the minimum configuration to run compiler

1.4. Purchase price of compiler if any (note: many unbundled manufacturers charge for compilers)

1.5. Annual maintenance cost if any. (If language is purchased from a third party then include estimated internal programming cost of providing support)

1.6. Estimated life of application

1.7. The first major guess! Each language for different thresholds for achieving syntacically correct compilers. Use internal knowledge, Auerbach, Datapro, consultants, etc.

2. APPLICATION

The application characteristics provide a difficulty factor for each language

APPLICATION	CHAR[8]	COBOL[2.1]	SPL
BUS/SCI			
BATCH/INTER			

2.1. Enter basic characteristic of application (B=business/S=science; B=batch/I=interactive)

2.2. Enter 1 if language manuals devote chapters to processing your application character
Enter 2 if there is an appendix
Enter 3 if there is a reference
Enter 4 if there is no mention

3. PORTABILITY

Portability is the capability of moving an application programmed in a specific language from your current computer to another manufacturer or even to another model of your current manufacturer with a minimum[10] of reprogramming. The table presents both a difficulty factor and a need factor (ie) your need for portability.

PORTABILITY	NEED[3.1]	COBOL[3.2]	SPL

3.1. Minimim is defined as 99% of compiler are clean

3.2. If you have no need for portability then skip or else
Enter 5 if you expect to upgrade in 1 year
Enter 4 if you expect to upgrade in 2 years
Enter 3 if you expect to upgrade in 3 years
Enter 2 if you expect to upgrade in 4 years
Enter 1 if you expect to upgrade in 5 years

3.3. Enter 1 if your language is a "standard" without sub or super sets
Enter 2 if a standard language sub or super set
Enter 4 if not a standard language

4. CONVERSION

Conversion in this paper is defined as converting your program specifications, flowcharts, etc., into the actual program. It is assumed that some form of structured programming is required.

CONVERSION	COBOL[4.1]	SPL

4.1. Enter 1 if language manuals devote chapter to structured programming
Enter 2 if there is appendix describing techniques
Enter 4 if there is a reference
Enter 8 if there is no mention

5. COMPILER SPEED

The actual language compiler must be analyzed to determine its relative speed. Again, this is a guess and it is recommended that internal studies or outside references be used

COMPILER SPEED	COBOL[5.1]	SPL

5.1. Enter 1 if fast one pass
Enter 2 if slow one pass or fast two pass
Enter 4 if slow two pass
Enter 8 if anything else

6. COMPILER AIDS

The number of aids available to the programmer can substantially reduce the programming effort. The table below represents some available aids.

AIDS	COBOL	SPL
EDITOR[6.1]		
COPYLIB[6.2]		
SEGMENTER[6.3]		
TOTAL AIDS		

6.1. Enter 1 if language or host computer has an excellent editor
Enter 2 if good
Enter 4 if fair
Enter 8 if none

6.2. COPYLIB is a COBOL facility that enables standard data divisions and/or procedure divisions to be incorporated into a program with one line of code
Enter 1 if available
Enter 8 if not available

6.3. Segmenter provide assistance in dividing program into logical segments for better program utilization and faster execution
Enter 1 if excellent
Enter 2 if good
Enter 4 if fair
Enter 8 if none

7. DEBUGGING

Languages provide various forms of debugging ranging from providing hex dumps to step by step traces.

DEBUGGING	COBOL[7.1]	SPL

7.1. Enter 1 if language reference devotes chapters to debugging
Enter 2 if there is appendix
Enter 8 otherwise

8. SELF-DOCUMENTATION

Self-documentation is somewhat of a myth. A specific language may provide the facility for self-documentation but without consistent enforced coding standards then any program can be obtuse. The table below rates only one variable, the naming of variables

DOCUMENTATION	COBOL[8.1]	SPL

8.1. Enter 1 if variable names can be actual words
Enter 2 if variable names can be reasonable abbreviations
Enter 4 if otherwise

9. PROGRAMMER AVAILABILITY

The popularity of a language has dictated the available supply of competent programmers in that language. A simple gauge of availability is to determine from want ads and/or an employment agency the average salary for a programmer with three years of experience in each language you are considering.

PROGRAMMER	COBOL[20]	SPL

9.1. Enter 1 for lowest salary
Enter 2 for salary 10% higher
Enter 4 for salary 20% higher
Enter 8 for salary 30% higher
Enter 16 if a salary is quoted at 40% or higher
Enter 32 if there is no quote

10. INTERFACES

Screen handlers, database, graphics, telecommunications, etc., packages are becoming increasingly necessary in computer installations. The ease of a language in interfacing with these packages is vital in any determination

INTERFACE	COBOL[10.1]	SPL
SCREEN		
DATABASE		
GRAPHICS		
10.2		

10.1. Enter zero if your application does not require else
Enter 1 if language manual devotes chapter
Enter 2 if appendix
Enter 4 if reference
Enter 8 if no mention

10.2. Skip if total is zero

11. SUMMARY

Fill in the table and choose your language

	COBOL	SPL
COST PER COMPILE		
APPLICATION[11.1]		
PORTABILITY		
CONVERSION		
COMPILER SPEEDS		
COMPILER AIDS		
DEBUGGING		
DOCUMENTATION		
PROGRAMMER		
INTERFACES		
TOTAL OF FACTORS		
LANGUAGE COST[11.2]		

11.1 The difficulty factors are added together to obtain "Total of Factors"

11.2. Language cost = cost per compiles * total of factors

12. POSTSCRIPT

The above is a simplified analysis of the factors for selecting a language. Other factors should be considered and different weights can be determined. However, the basic approach yields valuable insights and usually some interesting surprises!

INTERFACING QUANTITATIVE MODELING TECHNIQUES TO MFG 3000

JOHN A. STARK
HP3000 SYSTEMS ENGINEER/INDUSTRY SPECIALIST
HEWLETT-PACKARD COMPANY
SEATTLE, WASHINGTON

Hewlett-Packard Company is well-known in the electronic's industry as a manufacturer of computers and computer peripherals, components, instruments, microwaves, as well as many other products. Singling out the computer product line, we find software products that enhance the capability of the hardware. On the 3000, we find MPE the powerful operating system that controls and schedules online and batch processes. Additionally, there are several compilers, data management techniques, and general purpose utilities to aid the user in designing and implementing applications unique to his environment. During the 1980's Hewlett-Packard will continue to enrich its computer with new and/or improved products keeping in mind the basic philosophy of the company of not being a "ME TOO!" vendor. IE, developing products that will make a technological contribution to the marketplace. Expanding on this philosophy, some great marketeer (not to be confused with mouseketeer) asked "Why not combine our knowledge of computers and our knowledge of manufacturing and offer products that will aid other manufacturers in better controlling their environment? Let's develop something that is general purpose and can be used by anyone from a fruit processor to an aircraft manufacturer; something that is friendly and easy to implement and use; and let's take advantage of our existing hardware and software tools."

Business Systems Division was born with these goals in mind. Specifically, the areas that needed to be addressed were: Materials Management; Production Management; Order Management; Financial Management; and Sales Management. Because Materials Management yielded the quickest return on investment (by aiding in inventory reduction, thus reducing inventory carrying costs and freeing up working capital) and is the basic input to production, it was the logical place to start.

MFG 3000 is set of 3000 based application software, available from Hewlett-Packard, to aid in the planning for and control of raw materials. MFG/3000 consists of four modules which can be purchased separately or as an integrated package. The modules included in the current release of the software are Engineering Data Control (EDC), Inventory and Order Status (IOS), Material Requirements Planning (MRP), and Standard Product Costing (SPC). Basically, EDC is used to structure a Bill of Materials (i.e., outline what and how many parts go into the fabrication of a final assembly item). IOS keeps track of which of these detail parts are on order (whether purchased or fabricated in-house) and how many of each detail part is available in inventory for issue. MRP looks at a company's master schedule (the schedule of end items that the company is planning to build); looks at EDC to determine all of the component parts, the lead times and how many of each are required to build the end item; looks at IOS to determine if there are adequate detail parts in inventory or on order to fill the requirements generated by MRP; and finally, displays several reports with recommendations for appropriate action to be taken on each component part based on its lead time, due date, and order policy. (eg. new orders, orders to cancel, orders to expedite, etc.)

Purchase orders are released for those parts that are purchased from outside vendors and production control releases orders for those detail parts that are fabricated in-house. At the appropriate time, the completed detail parts are released to the shop and the finished product is assembled. Finally, Standard Product Costing is the means by which we are able to value our Bills of Material based on user-input accounting standards.

In short, we are able to reduce our inventory levels of raw material because we only order what we need, when we need it, versus stockpiling and hoping that we'll need it.

Sounds great! What also could a company possibly require to make its manufacturing process run smoothly and efficiently? Just purchase MFG 3000, load in your parts structure information, do a physical inventory count and load IOS, then run MRP and you'll know exactly what, when, and how much to order. Right? WRONG! We made several invalid assumptions that MFG 3000 does not address but must be considered in the overall manufacturing process.

First, we assumed that the Master Schedule miraculously appeared and will continue to reappear with each new planning horizon. Not a valid assumption! Someone has to create the Master Schedule. This is not a problem if items are put into the schedule only when an actual sales order is received. But what if we wanted to smooth out our production and try to predict over some planning horizon what the demand will be for our products? What tools does Hewlett-Packard offer in the area of forecasting? Currently, nothing. Quantitative methods, however, provides us with several models for forecasting demand based on past experience. One such model is Linear Regression. With this model, we are stating that there is a correlation between two or more variables, (i.e., one variable is dependent on one or more variables) and this relationship can be expressed in a linear fashion in the form $Y = A + BX + e$ where Y is the dependent variable, X is the independent variable, A is the Y intercept, B is $\frac{\Delta Y}{\Delta X}$ (slope) and e is the error term (the portion of Y's variance unexplained by X's variance.) The objective is to plot the "best fit" line between several points and use this line to predict the future. Figure 1 is a plot of past sales history. The Y axis is quantity sold and the X axis is product availability.

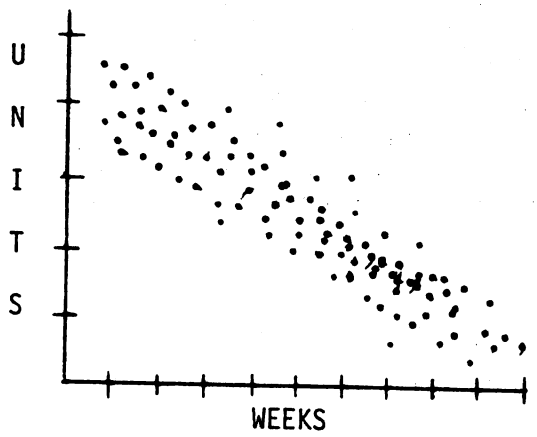


FIGURE 1

The method of determining the "best fit" is called Least Squares, which requires the simultaneous solving of two equations. These are:

$$\Sigma XY = A(\Sigma X) + B\Sigma X^2$$
$$\Sigma Y = nA + B\Sigma X$$

where Σ = summation
X = observations on X coordinate
Y = observations on Y coordinate
n = total pairs of observations

In our example the best fit line is Y = 24.51 - 1.63X.

Figure 2 is a graphic representation of this line. We can now readily predict the sales of our product simply by supplying an availability in weeks and solving the equation. For example, if we expect our product availability to be 13 weeks on the average over our planning horizon, then our demand will be 3.31 units.

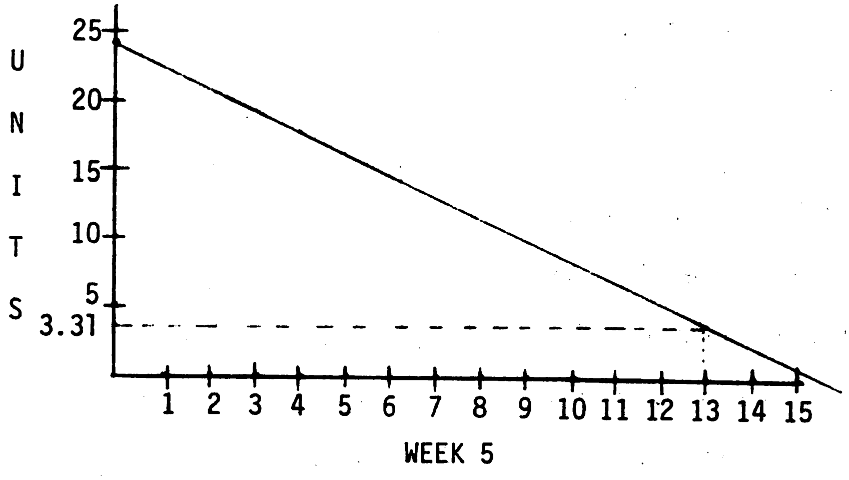


FIGURE 2

We have now established a master schedule for product #1 of our product line and continue in a similar fashion for the remainder of our goods.

This logic can even be applied to situations where demand is seasonal as can be seen in figure 3. In this situation, a line is fitted to a curve depicting cyclic demand for a product.

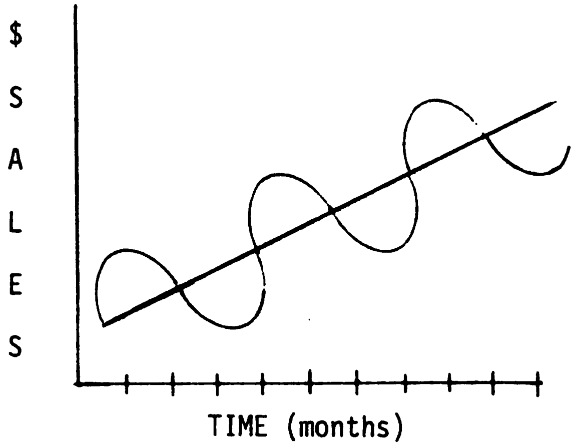


FIGURE 3

However, in many companies there isn't a correlation between a product and some variable (at least no quantifiable relationship). Stating that there is a linear relationship between sales and time in figure 4 would be nonsense. In this situation, a better technique to utilize would be

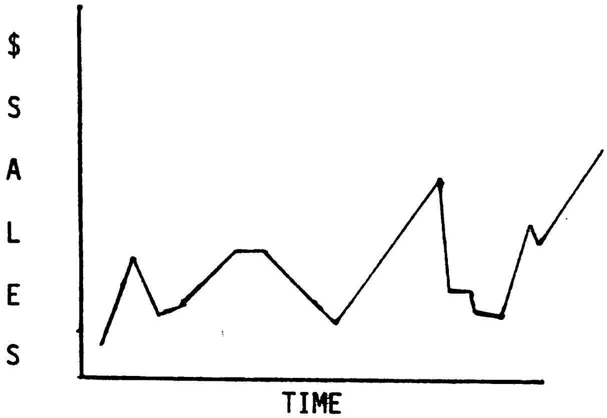


FIGURE 4

an exponential smoothing model. With this model we are able to forecast demand based on past experience but weighted using one or more smoothing constants to reflect any additional information that we might have. (i.e. sudden increase in sales due to some technological edge that we have over our competitors) and can be used for a steady increase or decrease in events.

The formula for an exponential smoothing model using one constant is:

$$\overline{F}_n = F_{n-1} + \alpha(D_n - F_{n-1}), \text{ or}$$

$$\overline{F}_n = D_n + (1 - \alpha) F_{n-1}$$

where: $\overline{F}_n$ = Forecasted Demand in next period
α = Smoothing constant ($0 < \alpha < 1$)
D_n = Actual Demand in current period
F_{n-1} = Forecasted Demand previous period

Essentially, what we are doing is predicting demand in the next period (n) based on what we predicted would happen in the current period (n-1), what actually happened and weighted by some constant .

For example, if we had experienced the following sales:

Period	Demand (Actual)	Forecast
1975	12	
1976	13	
1977	12	
1978	20	13
1979	?	15.8

and implemented this model at the end of 1977 (forecasting 13 units in 1978 assuming a smoothing constant of .4), then our forecast for 1979 would be

$$F_{1979} = F_{1978} + \alpha(D_{1978} - F_{1978})$$

$$F_{1979} = 13 + .4\,(20 - 13)$$

$$F_{1979} = \underline{15.8}$$

Using a larger value for α , say .8, demand for 1979 is 18.6 which says that we are putting an 80% weight on actual demand and a 20% weight on what we forecasted the demand to be in the previous period. The choice of α is normally guided by judgment though you could predict economically best or near best values through studies. This model can similarly be used for other products in our line and can be refined using triple smoothing constants of alpha, beta, and gamma.

Once we have done a reasonable job of forecasting demand, the next questions to ask are "do we have sufficient capacities to satisfy this demand? and is it profitable to make these products?", or - restated, given unit costs and selling price of my products and a finite factory capacity, "what optional mix of products should I build to maximize my profit subject to my demand constraints?".

Again, Hewlett-Packard currently does not address this area with a purchasable software product, but there are models available which will aid us in answering the questions posed.

One method that is commonly used is the "Business as Usual" approach. Others find sanctuary in the "S.W.A.G." model. Perhaps a better approach would be to utilize another technique from Quantitative Methods called Linear Programming. The object of this model is to maximize (eg. profit) or minimize (eg. costs) some function subject to a set of constraints. Consider the example of a firm that produces three products and has a planning horizon of one year. Table 1 is a summary of the firm's operating environment. To make one unit of product A it takes five machine hours and four assembly hours with each unit contributing $1.00 to profit.
($ PROFIT = $ SELLING PRICE - $ PRODUCTION COSTS)

Product	MH/Unit	LH/Unit	Demand	SP ($)	PC ($)	PM ($)
A	5	10	4	11	10	1
B	2	1	1	3	1	2
C	3	2	2	5	2	3

Total Machine Hour Capacity = 100 hours
Total Labor Hour Capacity = 80 hours
Total We Can Sell = 60 units

TABLE 1

Product A is the top end of the product line. It's a great product and unlike some of our other products, it works. We expect to sell at least four units. Product B is the low end of our line and is subject to frequent failures, but we expect to sell at least one unit during this planning horizon. Product C is the middle of our line and contributes the most to our profitability, so we'd like to sell a bunch. We have a bottleneck in our sales force in that they can only sell sixty units total of all products. There is one additional constraint imposed by the president of our company who states that "as long as I'm running the show, we'll never, I repeat NEVER, produce more than a total of five product B's! and/or product C's. I wanted to drop those darn things from the product line, but mom wouldn't let me."

The next step is to reduce table 1 to a set of equations which will be solved simultaneously. These are:

$5A + 2B + 3C \leq 100$	Machine Capacity Constraint
$4A + B + 2C \leq 60$	Sales Constraint
$B + C \leq 5$	President Constraint
$10A + B + 2C \leq 60$	Labor Capacity Constraint
$A, B, C \geq 0$	Nonnegativity Conditions
MAXIMIZE $A + 2B + 3C$	Objective Function

These equations can be solved manually using the SIMPLEX METHOD or very easily using your HP 3000 Computer. The final iteration (optional solution) is displayed in table 2. Note that some new variables, S_1 through S_4, were added. These are called Slack variables and are required to change the equations from inequalities to equalities. One Slack variable is required for each equation.

	A	B	C	S_1	S_2	S_3	S_4	Solution
S_1	0	- .5	0	1	0	-2	- .5	50
S_2	0	- .6	0	0	1	-1.2	- .4	22
C	0	1	1	0	0	1	0	5
A	1	- .1	0	0	0	- .2	.1	7
f	0	- .9	0	0	0	-2.8	- .1	

TABLE 2

Looking at the right most column we will find the optimal solution which says:

> Produce 7 units of A, 0 units of B, and 5 units of C for a total profit of $22.00 (7*$1 + 5*$3). Note that the firm has 50 hours of surplus machine time and 22 units of surplus sales capacity. The resource that was totally exhausted was the available labor hours.

Additionally, the solution tells us that for one more unit of labor we can expect a -10¢ increase in total profitability (see column S_4 Row F) and that we are not willing to pay for any additional machine time or sales capacity since we have excess. Also, note that if we decided to produce one unit of product B anyway, we would have to decrease production of product C by one unit resulting in an increase in machine hour surplus of .5 hours; an increase of available sales capacity of .6 units; an increase of product A production of .1 units and decrease in profitability of 90 cents. The model can be rerun relaxing or increasing the constraints to give the effect of a "what if?" model.

However, many firms have planning horizons that extend beyond one time period. For these firms, a static model may not be useful or may yield sub-optimal results. Fortunately, Quantitative Methods does provide models covering multiple time periods called Multi-Period Linear Programming or Dynamic Programming Models. For simplicity, consider the case of a company that produces a single product and is faced with making decisions over the next three time periods. Unit production costs are \$5, \$6, and \$6 for time periods 1, 2, and 3 respectively and must be paid in the period incurred. Storage costs of \$1 per unit per period must be paid in advance in the period the product is produced. Production capacity in period 1 is 40 units; in period 2, 50 units; and in period 3, 60 units. Unit selling prices are \$7, \$6.50, and \$8 for periods 1, 2, and 3 respectively. Sales capacity (demand) is 50 units in periods 1 and 2 and 70 units in period 3. The firm operates on a cash basis and has an opening balance of \$225. Cash from sales is received one period after the period in which the sale was made.

The objective is to maximize the profits subject to the constraints and realizing that products produced in one period may be sold in another. Let X_{ij} denote the product where i = period produced and j = period sold. Therefore, we end up with six unique combinations X_{11} (built in period 1, sold in period 1); X_{12}; X_{13}; X_{22}; X_{23}; X_{33} (assuming all products are sold at the end of period 3). Our contribution to profit then, for each period, is Selling Price - (Production Costs + Storage Costs). For period 1 it is \$7 - \$5 - 0. Our objective function for all periods thus becomes:

$$\text{MAXIMIZE} \qquad 2X_{11} + .5X_{12} + X_{13} + .5X_{22} + X_{23} + 2X_{33}$$

For the constraints the equations are:

Production

Period 1 $X_{11} + X_{12} + X_{13} \leq 40$

Period 2 $X_{22} + X_{23} \leq 50$

Period 3 $X_{33} \leq 60$

Sales

Period 1 $X_{11} \leq 50$

Period 2 $X_{12} + X_{22} \leq 50$

Period 3 $X_{13} + X_{23} + X_{33} \leq 70$

Cash

Period 1 $5X_{11} + 6X_{12} + 6X_{13} \leq 225$

Period 2 $-2X_{11} + 6X_{12} + 7X_{13} + 6X_{22} + 7X_{23} \leq 225$

Period 3 $-2X_{11} - .5X_{12} + 7X_{13} - .5X_{22} + 7X_{23} + 6X_{33} \leq 225$

The means of solving these equations would be identical to that of the pervious example except that I would recommend doing it via computer versus manually (unless, of course, you are a masochist).

The output table would also be similar to the previous example and would indicate how many to produce during each time period.

If the time periods that we are referring to are a year in length, this methodology could be further refined by using discounted cash flows as inputs to the model.

The outputs from this model can now become the inputs to our master schedule, but time phased to achieve the greatest profitability (or net present value if discounted cash flows are used). And as was previously mentioned, MRP can explode these master-scheduled items into the component parts and produce, as its output, reports of what, how much, and when to order the detail parts, and thereby reduce investments in inventory levels and minimize production disruptions due to stockouts.

Establishing a master schedule is only one of many problems that a firm must face. As the orders are released to the shops, the company may experience bottlenecks or queuing of parts waiting to go through a given workcenter. Perhaps a given type of machine in the workcenter is creating the bottleneck. A method for determining how many machines that you will need to eliminate the problem area is available with Queuing Theory (Waiting Line Models).

A modified version of the transportation model may be utilized to place or schedule orders on different machines taking into consideration the efficiency of each machine. Program Evaluation and Review Techniques (PERT) has a use in scheduling long lead time items. And the list goes on and on.

In summary, there are several modeling techniques available from Quantitative methods which will aid a company in better controlling its day to day operations and thereby increase its profitability. Even though Hewlett-Packard intends to address all areas of manufacturing from Materials Management to Capacity Planning, there is no need to wait for the one vendor solution. Utilize what has been available for years...Quantitative methods!

" MEMORY MANAGEMENT "

A VIDEOTAPE BY WENDELL HENRY

Extra Commentary by

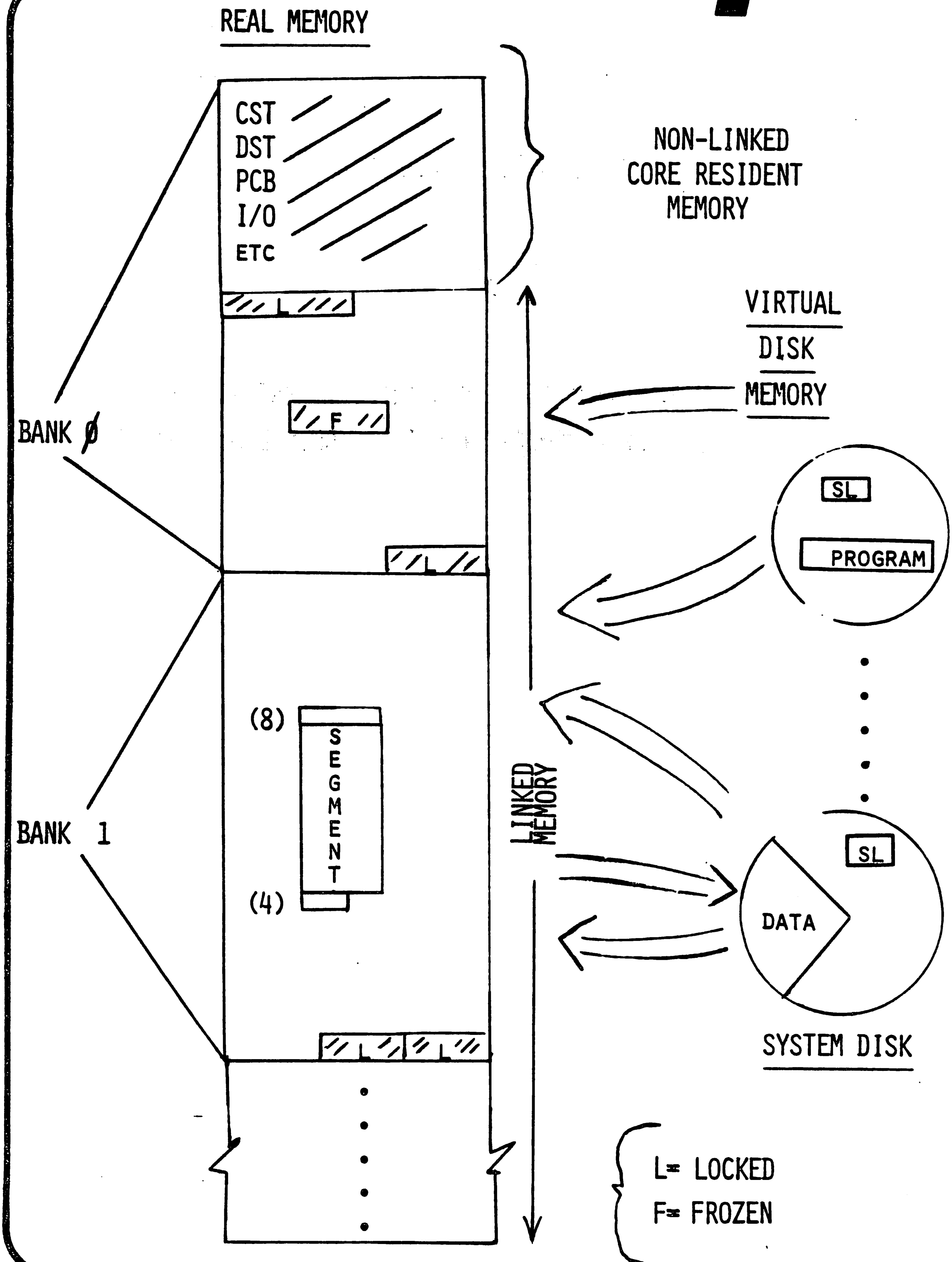
Chris Moeller
HP Cupertino

MPE III
MEMORY RESOURCE MANAGEMENT

MEMORY RESOURCE MANAGEMENT

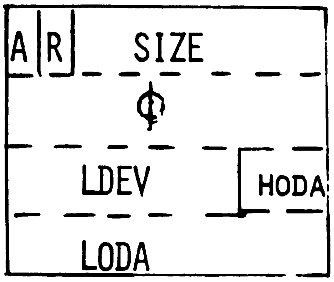
OBJECTIVE

ATTEMPT TO MAXIMIZE THE PROBABILITY THAT A SEGMENT WILL BE IN REAL MEMORY WHEN IT IS NEEDED BY A PROCESS.


REAL MEMORY
CST
DST
PCB
I/O
ETC
NON-LINKED CORE RESIDENT MEMORY
L
F
L
VIRTUAL DISK MEMORY
BANK 0
SL
PROGRAM
BANK 1
(8)
SEGMENT
(4)
LINKED MEMORY
SL
DATA
L
L
SYSTEM DISK
L= LOCKED
F= FROZEN

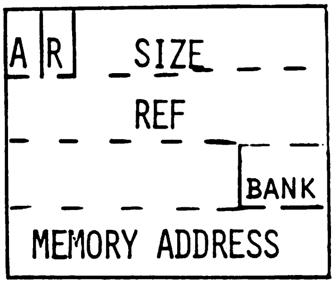
CST/DST ENTRY

- CST/DST ENTRY DEFINES MEMORY STATE OF SEGMENT


A
R
SIZE
Φ
LDEV
HODA
LODA

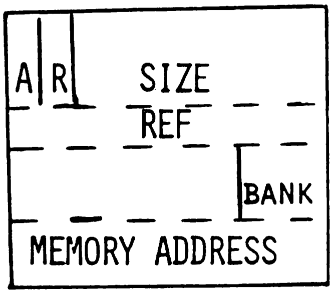
ABSENT SEGMENT

- A=1
- LOGICAL DEVICE # OF DISK
- HODA/LODA - DISK ADDRESS


A
R
SIZE
REF
BANK
MEMORY ADDRESS

PRESENT SEGMENT

- A=0
- REF - REFERENCE COUNT
- BANK/ADDR - BANK AND MAIN MEMORY ADDRESS


A
R
SIZE
REF
BANK
MEMORY ADDRESS

PSEUDO ABSENT SEGMENT

- A=1
- REF <> 0
- BANK/ADDR - BANK AND MAIN MEMORY ADDRESS

R - BIT DEFINES WHETHER SEGMENT HAS BEEN REFERENCED (PRESENT SEGMENT ONLY)

- SET BY MICROCODE DURING PCAL PROCESSING
- SET BY MIRCOCODE DURING MTDS/MFDS PROCESSING
- SET BY SOFTWARE DURING EXCHANGE DB

MEMORY STATES

- MEMORY IS MANAGED AS CODE AND DATA SEGMENTS
- AN ALLOCATED SEGMENT IS IN ONE OF THESE STATES
  - ABSENT — THE SEGMENT IS NOT IN MAIN MEMORY. IT IS IN VIRTUAL MEMORY (ON DISK)
  - PRESENT — THE SEGMENT IS IN MAIN MEMORY
    - AN UP-TO-DATE COPY OF A CODE SEGMENT ALSO EXISTS IN VIRTUAL MEMORY
    - AN OUT-OF-DATE COPY OF A DATA SEGMENT EXISTS IN VIRTUAL MEMORY
  - PSEUDO ABSENT — THE SEGMENT IS AVAILABLE FOR OVERLAY

MEMORY LINKS

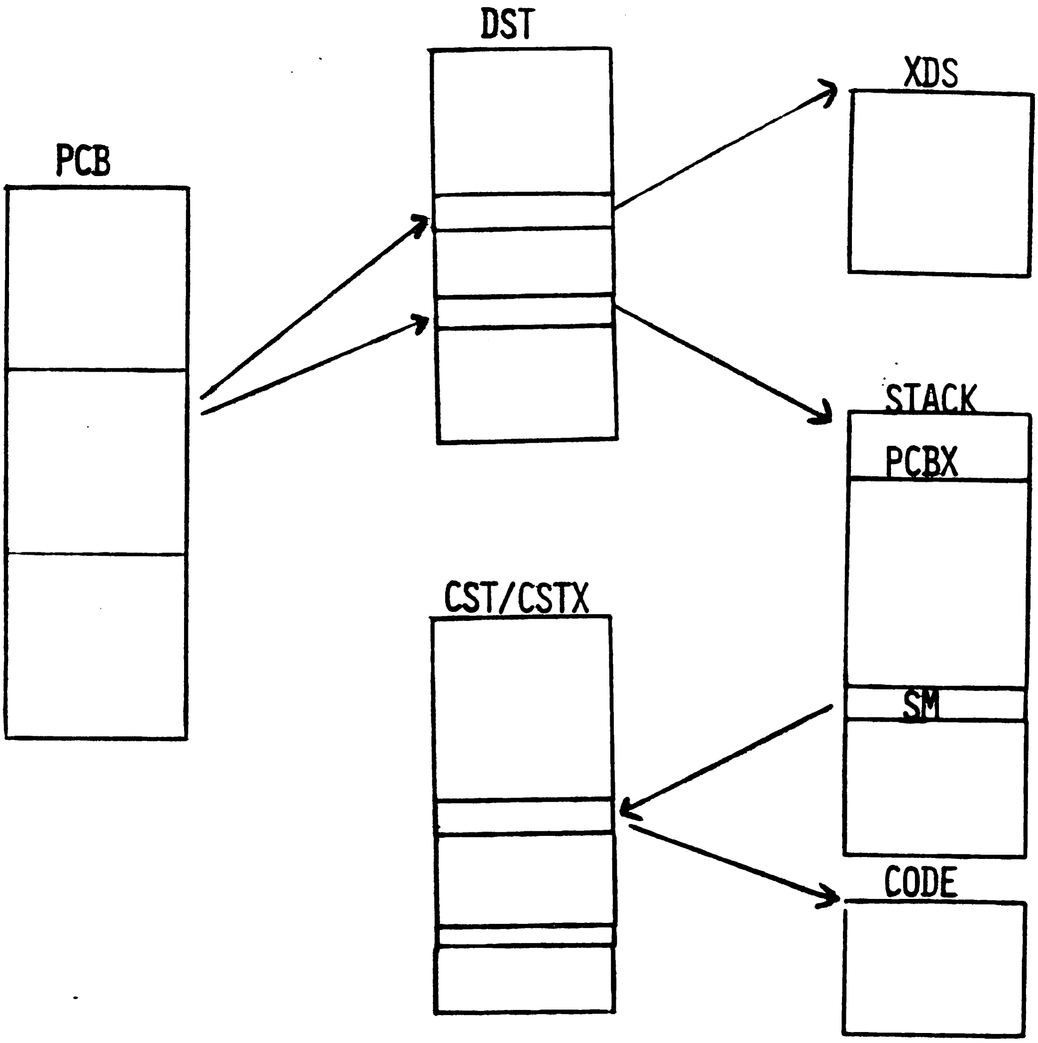
- ALL REGIONS OF LINKED MAIN MEMORY ARE PRECEDED AND FOLLOWED BY CONTROL INFORMATION (MEMORY LINKS)
  - MEMORY IS ALLOCATED AS MULTIPLES OF 4 WORDS
  - THE LINK IN FRONT OF AN AREA IS 8 WORDS LONG
  - THE LINE FOLLOWING AN AREA IS 4 WORDS LONG
- ALL FREE (UNALLOCATED) AREAS OF MAIN MEMORY ARE DOUBLY LINKED TOGETHER. THE MOST RECENTLY FREED REGION IS AT THE FRONT OF THE LIST
- ALL ALLOCATED AREAS OF MAIN MEMORY ARE POINTED TO BY THE CST/DST ENTRIES

PROCESS

- EXECUTABLE ENTITY
- CONTROL INFORMATION IN PCB
- PROPERTIES
    - HAS WORKING SET
    - HAS MEMEORY STATE
        - CORE RESIDENT
        - IN CORE
        - ABSENT
        - OUT OF MEMORY
    - ON SCHEDULING QUEUE
        - READY LIST - NEEDS THE CPU
        - DISCARD LIST - WAIT FOR EVENT COMPLETION BEFORE NEEDING THE CPU
        - NO QUEUE

PROCESS

- BASIC EXECUTABLE ENTITY
- SEVERAL PROCESSES MAY BE RUNNING ON BEHALF OF ONE USER (JOB)


DST
XDS
PCB
STACK
PCBX
CST/CSTX
SM
CODE

MEMORY ALLOCATION MANAGER (MAM)

- EXECUTES AS A PROCESS
- SCHEDULED BY DISPATCHER
    - DISPATCHER HAS DETECTED THAT THE HIGHEST PRIORITY PROCESS IS ABSENT
    - MAM WAS AWAKENED DIRECTLY BY ANOTHER PROCESS FOR HELP
- CONTROLS THE PLACEMENT/REPLACEMENT OF ALL SEGMENTS IN LINKED MEMORY

TABLES REQUIRED BY MAM

MTAB -	CORE RESIDENT - CONTROL INFORMATION: • FOR ALL CURRENT MAM I/O REQUESTS • FOR ALL CURRENT ABSENCE REQUESTS
AVAIL'-	CORE RESIDENT - THE CURRENT MEMORY AVAILABLE IN EACH BANK. AVAILABLE MEMORY IS THE SUM OF FREE AREAS AND AREAS ALLOCATED TO SEGMENTS WHICH ARE OVERLAY CANDIDATES
WSET -	CORE RESIDENT - CONTAINS THE WORKING SETS OF ALL EXECUTING PROCESSES
CHANGE'-	CORE RESIDENT - USED TO DETERMINE WHETHER AVAILABLE SPACE IN A BANK HAS INCREASED
ORDER'-	BANK NUMBERS SORTED IN ORDER OF MOST AVAILABLE SPACE

WORKING SET

- A WORKING SET PER EXECUTING PROGRAM
- ALL PROCESSES EXECUTING THE SAME PROGRAM SHARE THE SAME WORKING SET - POINTED TO BY PCB
- ALL "PRESENT" SEGMENTS MUST BE MEMBERS OF AT LEAST ONE WORKING SET
- A "SYSTEM" WORKING SET IS MAINTAINED FOR ALL MPE CODE AND DATA SEGMENTS SINCE THESE SEGMENTS ARE SHARED BY ALL EXECUTING PROGRAMS
- EACH WORKING SET IS DYNAMIC IN SIZE
- WHEN A SEGMENT IS REMOVED FROM A WORKING SET IT IS DEALLOCATED OR MADE AN OVERLAY CANDIDATE

WORKING SET

- INDICATES WHICH CODE AND DATA SEGMENTS ARE USED BY THE EXECUTING PROGRAM MOST FREQUENTLY
- IF AN EXECUTING PROCESS IS FAULTING FREQUENTLY THE WORKING SET IS ALLOWED TO GROW
- IF AN EXECUTING PROCESS IS FAULTING INFREQUENTLY THE WORKING SET IS REDUCED
  - IF THE LAST FAULT WAS < 100MS AGO THE NEW SEGMENT IS ADDED TO THE WORKING SET
  - IF THE LAST FAULT WAS > 100MS AGO ALL SEGMENTS WHICH WERE NOT REFERENCED DURING THE LAST 100MS ARE REMOVED FROM THE WORKING SET. THE NEW SEGMENT IS ADDED.

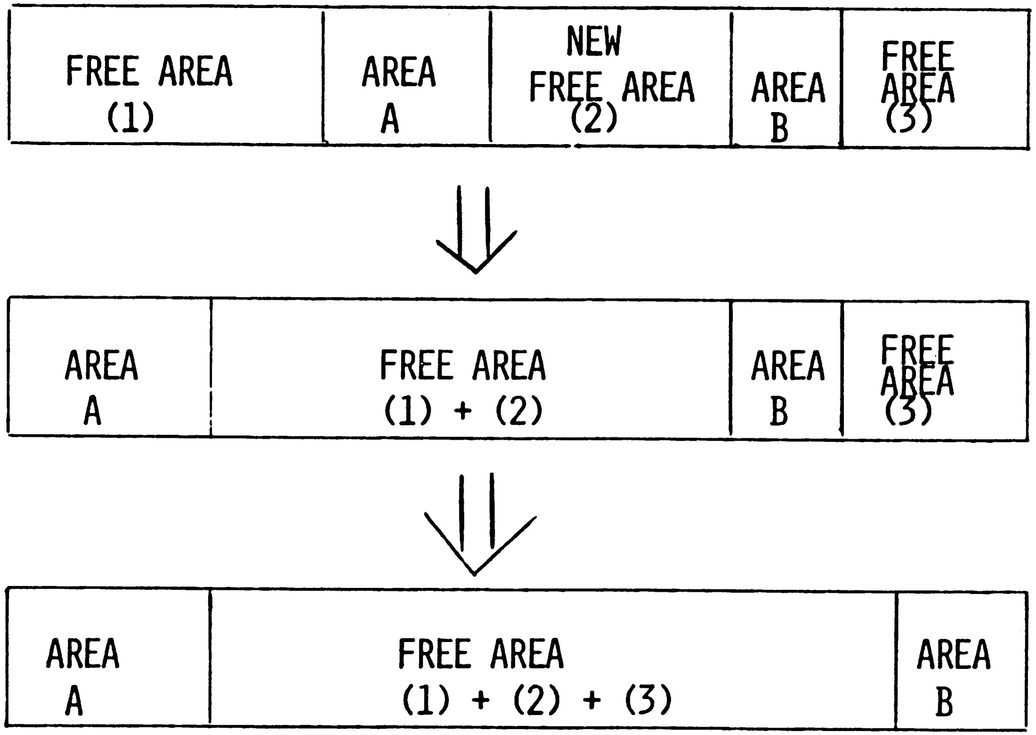
WORKING SET PARAMETERS

ALFA - TIME SINCE LAST REFERENCE COUNTER UPDATE. STORED IN EACH WORKING SET. EACH TIME A PROCESS EXECUTES ITS CPU USAGE WILL BE ADDED TO ALFA FOR HIS WORKING SET.

TAU - A VALUE OF 100 MS. PRESENTLY CONSIDERED TO BE AN OPTIMUM INTER-FAULT TIME

LAT - TIME SINCE LAST ABSENCE TRAP FOR A PROCESS. STORED IN EACH PROCESS' PCB.

FREE SPACE COMPACTION

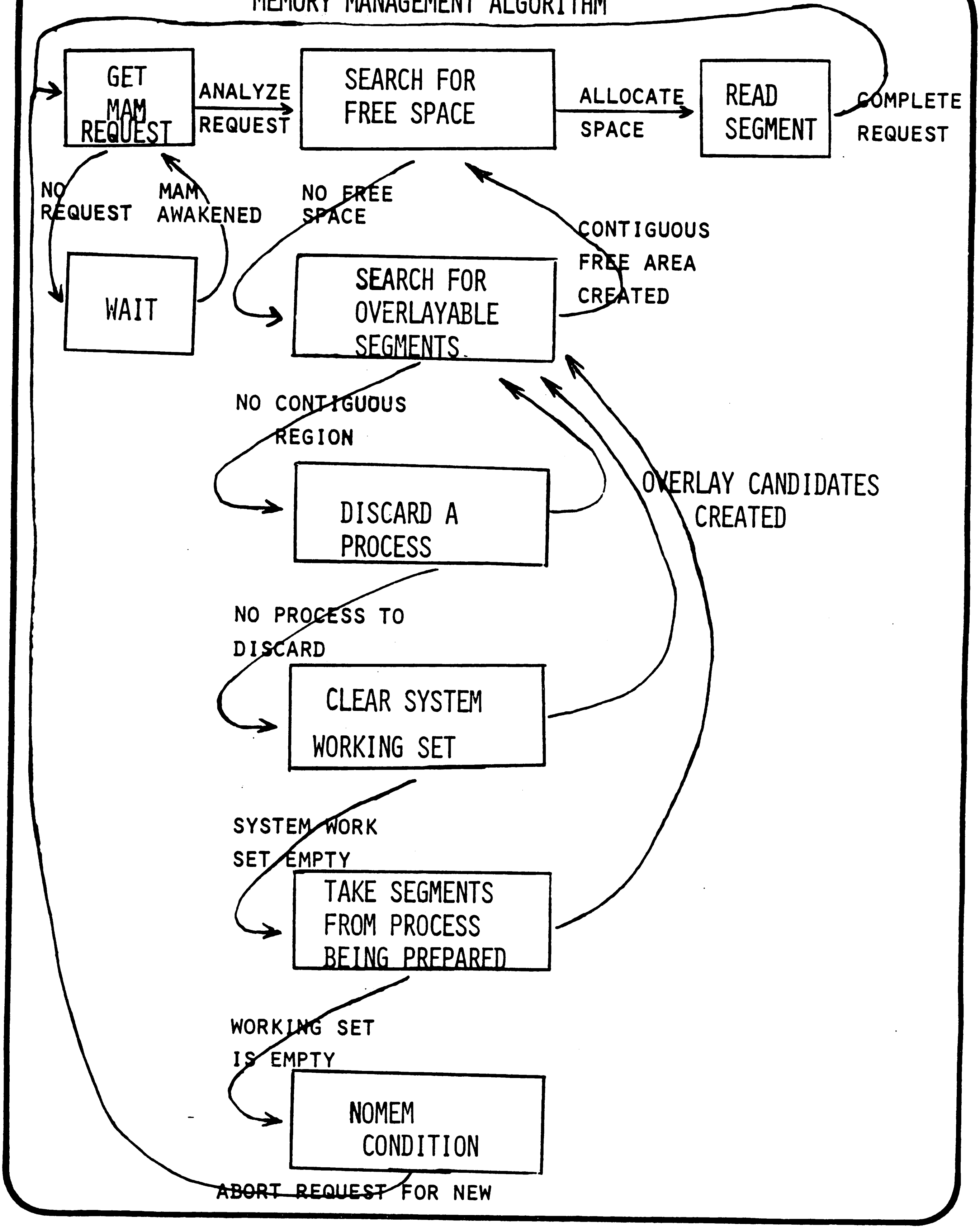


FREE AREA (1)	AREA A	NEW FREE AREA (2)	AREA B	FREE AREA (3)

AREA A	FREE AREA (1) + (2)	AREA B	FREE AREA (3)

AREA A	FREE AREA (1) + (2) + (3)	AREA B

MEMORY MANAGEMENT ALGORITHM



GET MAM REQUEST

ANALYZE REQUEST

SEARCH FOR FREE SPACE

ALLOCATE SPACE

READ SEGMENT

COMPLETE REQUEST

NO REQUEST

MAM AWAKENED

WAIT

NO FREE SPACE

SEARCH FOR OVERLAYABLE SEGMENTS

CONTIGUOUS FREE AREA CREATED

NO CONTIGUOUS REGION

DISCARD A PROCESS

OVERLAY CANDIDATES CREATED

NO PROCESS TO DISCARD

CLEAR SYSTEM WORKING SET

SYSTEM WORK SET EMPTY

TAKE SEGMENTS FROM PROCESS BEING PREPARED

WORKING SET IS EMPTY

NOMEM CONDITION

ABORT REQUEST FOR NEW

CONFERENCE PARTICIPANTS

HPGSUG 8TH INTERNATIONAL MEETING   ALHABETIC LIST OF ATTENDEES   10/24/79   PAGE 1

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
ABEHASSERA	ROBERT		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
AGAESSE	REMY		SOCIETE BECKMANN	52 CH. DES BOURDONS	GAGNY		FRANCE
ALFERT	STANLEY H.	JOINT ACTUARY	GRESHAM LIFE ASSURAN	2/6 PRINCE OF WALES	BOURNEMOUTH	BH4 9HD	U.K.
ALMKVIST	DAG		AB BOFORS	POST BOX 900	BOFORS	69020	SWEDEN
ANDERSON	CARL W	MKT. MGR. COMP. SUPP	HEWLETT PACKARD	1330 KIFER ROAD	SUNNYVALE	CA95051	USA
ANTO	JOHN M	SYSTEMS ANALYST	ILEC	120 S.RIVERSIDE PLAZ	CHICAGO (ILL)	60606	USA
ARMSTRONG	JACQUES						
ASTROM	FOLKE		AB BOFORS	POST BOX 900	BOFORS	690 20	SWEDEN
AYRAS	JAAKKO		JAAKO POYRY GROUP	KAUPINTIE 3	HELSINKI 40	SF-00400	FINLAND
BAILLEUX	MARCEL		HACHETTE	79 BD ST GERMAIN	PARIS	75006	FRANCE
BARK	EVERT	EDP MANAGER	NESTLE NEDERLAND B.V	HILVERSUMSTRAAT 316	AMSTERDAM	1024 KG	NEDERLAND
BARLOW	MIKE	SALES DEVELOPMENT EN	HEWLETT PACKARD GMBH	HERRENBERGER STR 110	BOEBLINGEN	7030	GERMANY
BARTHE	MR	DIRECTEUR GENERAL	AQUITAINE SYSTEMES	6 RUE PAUL BERT	PUTEAUX	92800	FRANCE
BASTIEN	MR		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
BAYLE	R.		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
BEAUVILLAIN	KLEBER	GENERAL MANAGER	HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
BENEDEK	CSABA		SECRETARIAT AGA KAHN		GOUVRIEUX		FRANCE
BENOIT	WAYNE F	MGR PRODUCT DEVLPMT	EPSILON DATA MGMT	24 NEW ENGLAND EXEC	BURLINGTON	MA01803	USA
BERGMANN	FRITZ	HEAD PROG.DEPT	UNILEVER SCHWEITZ AG	FOERRLIBUCKSTR.10	ZUERICH	CH 8031	SWITZERLAND
BERGSTROM	WILLY	DP-MGR	ELKRAFT,P-EDB	PARALLELVEJ 19	LYNGBY	DK 2800	DENMARK
BERNALIN	J.F	INGENIEUR	HACHETTE	79 BLD ST GERMAIN	PARIS	75006	FRANCE
BERTHAUD	JEAN-LUC	PROFESSEUR	ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
BERTHONNEAU	PIERRE	INGENIEUR ANALYSTE	C.A.M.I.F.	TREVINS DE CHAURAY	NIORT CEDEX	79039	FRANCE
BEVERINA	CHRISTIAN	ASSISTANT TECHNIQUE	ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
BIRKWOOD	ILENE M	SE MANAGER	HEWLETT PACKARD	19447 PRUNERIDGE AVE	CUPERTINO CA	95118	USA
BLASSER	HEINER	MKTG MGR COMPUTERS	HEWLETT PACKARD SA	7 RUE DU BOIS DU LAN	MEYRIN GENEV	CH 1217	SWITZERLAND
BLUCERT	STEPHEN A.	THIRD PARTY SPECIALI	HEWLETT PACKARD LTD	KING STREET LANE	WINNERSH	B	U.K.
BOND	ROBERT T.	MARKETING MANAGER	HEWLETT PACKARD	19447 PRUNERIDGE AVE	CUPERTINO	CA95014	USA
BONTOUT	ROGER		SOCIETE PARKER		PARIS		FRANCE
BORG	JEAN-PIERRE		TECHNIKUM WINTERTUR	TECHNIKUMSTR. 9	WINTERTHUR	CH-8401	SWITZERLAND
BOS	JAAP		HEWLETT PACKARD		AMSTERDAM		NEDERLAND
BOUCARD	BERNARD	ANALYSTE-SYSTEME	C.A.M.I.F.	TREVINS DE CHAURAY	NIORT CDEX	79039	FRANCE
BOULET	M.FRANCE		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
BOURLIER	MAURICE	PDG SERIC	SERIC	120,RUE DE JAVEL	PARIS	75015	FRANCE
BRAENDLI	EMIL R	DR	TECHNIKUM WINTERTHUR	TECHNIKUMSTRASSE 9	WINTERTHUR	CH-8401	SWITZERLAND
BREAUD	LAURENCE		HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
BREON	DICK E	PRINCIPAL CONSULTANT	TASCO INC	221 PINE ST.	SAN FRANCISCO	94104	USA
BROUWER	PETER	SYSTEEMONTWERPER	U.N.U.VERKOOPGROEP	DE BOELELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
BROWN	DALE A	DIR OF INSTL RESEARC	MARY WASHINGTON CLG	PO BOX 1081 CLG STA	FREDERICKSBUR	22401	USA
BROWN	DAVID	PRESIDENT	NICE CORPORATION	4357 SOUTH AIRPORT P	OGDEN UTAH	84403	USA
BRUN	MARC		HEWLETT PACKARD		GRENOBLE		FRANCE
BRYDEN	WILLIAM		BOARD OF HPGSUG				USA
BURGER	RICHARD		CLAYBROOK COMPUTING	70 BROOK STREET	LONDON W-1		U.K.
BYE	PER	SYSTEMS ENG. MANAGER	HEWLETT PACKARD NORG	OSTERNOALEN 18	OSTERAS	1345	NORWAY
CANTWELL	FRANK E	DIRECTOR PRODUCT DEV	EPSILON DATA MGMT	24 NEW ENGLAND EXEC	BURLINGTON	M01803	USA
CARNES	LANCE A	PROJECT MANAGER	GENTRY INC	609 KEARNEY STREET	EL CERRITO	CA94530	USA
CASSIER	DOMINIQUE		ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
CASTEEL	MICHAEL A	PARTNER	COMPUTING CAPABILITY	1101 SAN ANTONIO ROA	MNT VIEW CALI	94043	USA
CHADWICK	GRAHAM		DE ZOETE & BEVAN	25 FINSBURY CIRCUS	LONDON EC2		U.K.
CHARD	MILES M	DATA PROCESSING MGR	CLA-VA CO	RPO BOX 1325	NEWPORTBEACH	92663	USA
CHARRIER	MR		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
CHOLLET	ROLAND	SALES MANAGER	HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
CHRISTIANSSON	STURE		AB BOFORS	POST BOX 900	BOFORS	69020	SWEDEN
CIMRING	RON		ANMERCO SA COMPUTER	41 FOX STREET	JOHANNESBURG	2001	SOUTH AFRICA

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
CLARKE	PETER		SCHOOL OF SIGNALS		BLANFORD		U.K.
COCCONCELLI	MARIO	EDP MANAGER	ABBIGLIAM	VIA F.LLI CERVI 61/A	REGGIO EMILIA	42100	ITALY
COHEN	JO	JOURNALIST	01 INFORMATIQUE				FRANCE
COLDREN	J DAVID	CRIMINAL JUSTICE DIR	IL LAW ENFORCEMENT C	120 S.RIVERSIDE PLAZ	CHICAGO ILL	60606	USA
COLLINS	LEO W	SYSTEMS MANAGER	OMEGA SYSTEMS INC	3720 S ROCKWELL ST	CHICAGO ILL	60632	USA
COOPER	ROY	SYSTEMS ENG.MANAGER	HEWLETT PACKARD LTD	KINGSTREET LANE	WINNERSH BERK		U.K.
CORDELLE	YANN		COGELOG	1 RUE DES QUINCONCES	GIF SUR YVETT	91190	FRANCE
CROSBY	DONALD I	CUSTOMERS RELATIONS	HEWLETT PACKARD	19447 PRUNERIDGE AVE	CUPERTINO CA	95014	USA
DAHAN	J.PIERRE		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
DALLA PRIA	GASTONE	SYSTEM ENGINEER COOR	HEWLETT PACKARD	VIA G.DI VITTORIO 9	CERNUSCO S/N	20063	ITALY
DAMM	JACK A.	PRINCIPAL	THE PALO ALTO GROUP	790 LUCERNE DRIVE	SUNNYVALE	CA94086	USA
DANG	JOSEPH		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
DAVIES	IVOR	DATA PROCESSING MANA	FIELD SONS & CO LTD	LIDGET GREEN	BRADFORD	BD7 2RQ	U.K.
DE CARLO	GIANNI	SYSTEM ENGINEER MANA	HEWLETT PACKARD	VIA G.DI VITTORIO 9	CERNUSCO S/N	20063	ITALY
DE PRATO	GIOVANNI		INFOTER SRL	VIA SAGITTARIO 54	MODENA		ITALY
DE ROOS	HENK		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
DE STE MARESV	PHILIPPE		HEWLETT PACKARD		GRENOBLE		FRANCE
DEBRUN	MR.		STE SONACA		GOSSELIES	B-6200	BELGIUM
DEHASSE	MR.		STE SONACA		GOSSELIES	B-6200	BELGIUM
DEMONCEAUX	HENRI	SEO MANAGER	HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
DENNY	JAY	WORLDWIDE SEO MANAGE	HEWLETT PACKARD	7 RUE DU BOIS DU LAN	MEYRIN GENEVA	1217	SWITZERLAND
DENT	RICHARD A	MANAGER,HP3000 GROUP	MICROWAVE ASSOCIATES	SOUTH AVE.,BLDG #3	BURLINGTON	M01803	USA
DENYS	BERNARD		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
DIETZGEN	JOE	PRODUCT MGR HP300	HEWLETT PACKARD GMBH	PO 1430	BOEBLINGEN	D7030	GERMANY
DOLMANS	PIERRE		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
DOWLING	JIM	OPERATIONS MANAGER	BOSE CORP	100 MOUNTAIN RD.	FRAMINGHAM (M		USA
DOYER	ANDRE	RESPONSABLE INFORMAT	SA M GUETAT	ZI BD MONGE	MEYZIEU	69330	FRANCE
DREILING	CHERYL		LAWRENCE LIVERMORE L		LIVERMORE (CA		USA
DREILING	LESLIE		LAWRENCE LIVERMORE L		LIVERMORE (CA		USA
DRYNAN	GIL		BOARD OF HPGSUG				USA
DUBACH	JAN D	SYSTEEMONTWERPER	U.I.N.U. VERKOOPGROE	DE BOELELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
DUSSART	LUC		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
EATON	JOHN		BOARD OF HPGSUG				U.K.
EBERSOLD	JOHANNES	DR	TECHNIKUM WINTERTHUR	TECHNIKUMSTRASSE 9	WINTERTHUR	CH-8401	SWITZERLAND
EDWARDS	JON		JENNISON ASSOCIATES	270 PARK AVE	NEW YORK	10017	USA
EEROLA	ANNELE	M. SC.	JAAKO POYRY CONSULT.	PL 16	HELSINKI 40		FINLAND
EKELOF	GUNNAR	SALES ENGINEER	HEWLETT PACKARD AB	BOX 20502	STOCKH	OS-161 20	SWEDEN
ELVSAAS	BJORN	ENGINEER	BERGEN INGENIORHOGSK	LARS HILLESGT.34	BERGEN	5000	NORWAY
ENDLISS	PHILIP I	DISTRICT SALES MANAG	HEWLETT PACKARD	P.O. BOX 92105	LOS ANGELES	C90009	USA
ENDTINGER	ERIC	SYSTEM ENGINEER	HEWLETT PACKARD AG	ZURCHERSTRASSE 20	SCHLIEREN	8952	SWITZERLAND
ERIKSSON	BJORN	SYSTEM ANALYST	AB TRAV OCH GALOPP	BOX 1733	STOCKH	OS-11187	SWEDEN
EYSSERIC	BRIGITTE		ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
FAIRCHILD	JAMES B	SUPERVISOR OF OPNS	DOMTAR CHEMICALS	395 DEMAISONNEUVE W.	MONTREAL	QUH3A 1L6	CANADA
FEU	STEVE	AREA SE MANAGER FOR	HEWLETT PACKARD	7 RUE DU BOIS DU LAN	MEYRIN GENEVA	1217	SWITZERLAND
FLEISHMANN	WERNER M.	DIPL.MATH	HALBERGERHUETTE GMBH	SAARBRUECKER STR.	SAARBRUECKEN-D-6604		GERMANY
FLOYD	TERRY II	INFOSYSTEMS MANAGER	THERMON MANUFACTURIN	100 THERMON DRIVE	SAN MARCOS	TX78666	USA
FOIADELLI	ARMANDO	SALES SUPERVISOR	HEWLETT PACKARD AG	ZURCHERSTRASSE 20	SCHLIEREN	8952	SWITZERLAND
FOURCADE	RENE	RESPONSABLE SERVICE	CARCEPT	1 RUE JACQUES BINGEN	PARIS CEDEX	175825	FRANCE
FOX	STEPHEN L	D P MANAGER	JOHNSON HILL PRESS	1233 JANESVILLE AVE	FT ATKINSON	W53538	USA
FUGGITT	CHRIS	R&D SECTION LEADER	HEWLETT PACKARD GMBH	HERRENBERGERSTRASSE	BOEBLINGEN	7030	GERMANY
GAISER	HANS GEORG	SYSTEM-PROGRAMER	HEWLETT-PACKARD GMBH	HERRENBERGER STR.110	BOEBLINGEN	D-7030	GERMANY
GAREL	MICHEL	SALES MANAGER	HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
GARZA G.	GUILLERMO	ASSISTANT MGR EDP	INDUSTRIAL MIN.MEXIC	BAJA CALIFORNIA 200	MEXICO 7 DF		MEXICO
GASPERI	MAURIZIO	PRODUCT MANAGER	HEWLETT PACKARD GMBH	HERRENBERGERSTR. 130	BOEBLINGEN	7030	GERMANY

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
GLOSS	GREGORY	COBOL PROJECT MANAGE	HEWLETT PACKARD	19447 PRUNERIDGE AVE	CUPERTINO CA	95014	USA
GOLDBERG	MICHAEL C	PRESIDENT	FINANCIAL DATA	2670 TIGERTAIL AVE	MIAMI FL	33133	USA
GOLDMAN	RICHARD	DIR SOFTWARE DEVELOP	HARLYN PRODUCTS	1515 S. MAIN ST.	LOS ANGELES (	90015	USA
GOMEZ J	LUIS	MANAGER EDP	INDUSTRIAL MIN. MEXI	BAJA CALIFORNIA 200	MEXICO 7 DF		MEXICO
GOURIOU	HERVE	SYSTEM ENGINEER	HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
GRAHAM	JEFF I	SALES ENGINEER	HEWLETT PACKARD LTD	KING STREET LANE	WINNERSH WOKI	BERKSHIRE	U.K.
GRANDCHAMP	FRANCOIS	ASSISTANT	ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
GRANGE	DAVID	SYSTEM DESIGNER	W.H.SMITH & SON LTD	GREENBRIDGE ESTATE	SWINDON	SN3 3LD	U.K.
GREEN	GARY		BOARD OF HPGSUG				USA
GREEN	ROBERT M	PRESIDENT	ROBELLE CONSULTING	*130-5421 10TH AVENU	DELTA BC	V4M 3J9	CANADA
GRIM	JELLE	SYSTEM MANAGER	ROYAL BOS KALIS WEST	ROSMOLENWEG 20	PAPENDRECHT		NEDERLAND
GRIVEL	MICHEL	DIRECTEUR CFP	ESCL	23 RT DE DARDILLY	ECULLY	69---	FRANCE
GROSSLER	JORG	DIPL.-ING	PROZESSRECHNERVERBUN	SEKR.:HE3,EINSTEINUF	BERLIN 10	D-1000	GERMANY
GUELTON	JACQUELINE		HEWLETT PACKARD		GRENOBLE		FRANCE
GUERIN	DIDIER	MARKETING MANAGER	HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
GUEX	MARTIAL W	ENGINEER	NESTEC	POST BOX 88	LA TOUR-DE-PECH	1814	SWITZERLAND
GUIGOZ	MR		HEWLETT PACKARD	15 RUE DU BOIS DU LA	MEYRIN GENEVA		SWITZERLAND
GUJER	GUENTHER	ING.HTL	ALU.AG MENZIKEN		MENZIKE	5737	SWITZERLAND
HAMPTON	JEAN K	PROGRAMMER	WM.C.BROWN CO	2460 KERPER BLVD	DUBUQUE IA	52001	USA
HEBERT	CHRISTIAN	SYSTEM ENGINEER	HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
HEDA	SHARAD		BOARD OF HPGSUG				USA
HEIMRICH	FRANZ	INTERN.EDP MANAGER	FESTO MASCH.FABRIK	RUITER STR.86,PF6040	ESSLINGEN 1 BD	7300	GERMANY
HEINING	MR		HEWLETT PACKARD GMBH	HERRENBERGER STRASSE	BOEBLINGEN	7030	GERMANY
HERRIGER	HANS	MARKETING SUPPORT MA	HEWLETT-PACKARD S.A.	7 RUE DU BOIS DU LAN	MEYRIN GENE	1217	SWITZERLAND
HINES	RELLA		BOARD OF HPGSUG				USA
HING	TREVOR J	SENIOR BUSINESS ANAL	W.H.SMITH & SON LTD	GREENBRIDGE ESTATE	SWINDON WILTS	SN3 3LD	U.K.
HINRICHS	UWE	SYSTEM ENGENEER SUPE	HEWLETT PACKARD GMBH	VERTRIEBSBURO KAPSTA	HAMBURG 60	2000	GERMANY
HISSINK	BERT		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
HODSON	N.B		CIE JOHN HOSKYNS	ASHTON LANE	SALE CHESHIRE	M33 1NP	U.K.
HOFF	MARC		HEWLETT PACKARD	PRUNERIDGE ROAD	CUPERTINO CA	94560	USA
HOFMANN	JUERG F	SYSTEM MANAGER	NESTEC CO LTD	CASE POSTALE 88	LA TOUR DE PE	1814	SWITZERLAND
HOLLOWS	ANTHONY	PROJECT MANAGER	CROWN DEC. PRODUCTS	CROWN HOUSE HOLLINS	DARWEN LANC	BB3 0B9	U.K.
HOLOPAINEN	PENTII		RAHA-AUTOMAATTTIYH.	MARIANKATUS	HELSINKI	00170	FINLAND
HULST	JAN CA	MANAGER EDP	UNILEVER-EMERY NV	BUURTJE 7	GOUDA		NEDERLAND
HUNGER	WOLFGANG	EDP MANAGER	PROGNOS A.G.	VIADUKSTRASSE 65	BASEL	CH-4011	SWITZERLAND
HVAMB	WIDAR		STATE TECH. COLLEGE		KONGSBERG ING		NORWAY
IUPPA	DAVID R.	SALES DEVELOPMENT MG	HEWLETT PACKARD GMBH	HERRENBERGERSTR. 130	BOEBLINGEN	7030	GERMANY
JACOBSON	VICTOR S	EDP MANAGER	GLAXO HOLDINGS LTD	63 GRAHAM STREET	LONDON	NI	U.K.
JEGARD	YVES		HEWLETT PACKARD		GRENOBLE		FRANCE
JOALLAND	GERARD	ENGINEER	CREDIT LYONNAIS	10 RUE ST AUGUSTIN	PARIS	75002	FRANCE
JUERN	FRITZ H.	SEO-MANAGER	HEWLETT-PACKARD GMBH	BERNERSTR. 117	FRANFUR	D6000	GERMANY
JOHNSON	CHARLES F	DATA PROCESSING MGR	FERRIS BUSSCHER&LOHM	339 E.16TH STREET	HOLLAND MI	49423	USA
JOHNSON	GARY		DOMTAR CHEMICAL				USA
JOHNSON	GERRY	TECHNICAL CO-ORDINAT	DOMTAR CONSTRUCTION	2001 UNIVERSITY ST.	MONTREAL QUE	H3A 2A6	CANADA
JOHNSON	RAY E.	SERVICE ENGINEERING	HEWLETT PACKARD	19447 PRUNERIDGE AVE	CUPERTINO CA	95014	USA
JOLLIET	MR		BANQUE CANTONALE VAU	PLACE ST FRANCOIS	LAUSANNE	CH-1000	SWITZERLAND
JONES	KEITH J	DATA PROCESSING MANA	GRESHAM LIFE ASSURAN	2/6 PRINCE OF WALES	BOURNEMOUTH D	BH4 9HD	U.K.
JONES	THOMAS		EPSILON		BURLINGTON (M		USA
JONSSON	MATS	SENIOR SYSTEM ENGINE	HEWLETT PACKARD AB	BOX 20502	STOCKHO	S-161 20	SWEDEN
JORDANS	WILLEM A	SYSTEM-ANALIST	NESTLE NEDERLAND B.V	HILVERSUMSTRAAT 316	AMSTERDAM	1021024 LG	NEDERLAND
JURACK	HEINZ	COMMUNICATIONS SPECI	HEWLETT PACKARD GMBH	HERRENBERGERSTR. 130	BOEBLINGEN	7030	GERMANY
KAESSMAYER	HANS	SYSTEM ENGINEER	HEWLETT PACKARD				AUSTRIA
KAHAN	JOHAN	DR	KLIN KEM LAB	SOUTHOSPITAL	STOCKHOLM	S-10064	SWEDEN
KALTBRUNNER	HANS	DIPL.ING.	HABEGGER AG	INDUSTRIESTR. 2	THUN	CH-3601	SWITZERLAND

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
KASSUM	SULTAN	COMPUTER MANAGER	H.H.THE AGA KHAN'S S	AIGLEMONT	GOUVIEUX	60270	FRANCE
KELLOGG	MR		HEWLETT-PACKARD GMBH	HERRENBERGER STR.110	BOEBLINGEN	D-7030	GERMANY
KENTON	RALPH H	MGR SOFTWARE DISTRIB	HEWLETT PACKARD	19310 PRUNERIDGE AVE	CUPERTINO CA	95014	USA
KHURSH	AHMED	SYSTEMS MANAGER	MCMASTER UNIVERSITY	1200 MAIN ST W RM2D5	HAMILTON ONT	L8S 4J9	CANADA
KIISKINEN	JOUNI		PTT OF FINLAND		HELSINKI		FINLAND
KLEIN	KLAUS DIETE	SYSTEM ENGINEER	HEWLETT PACKARD GMBH	HERRENBERGER STRASSE	BOEBLINGEN	7030	GERMANY
KLEIST	SVEN A.	CIVING	ATLAS COPCO MCT AB		STOCKHOLM	10260	SWEDEN
KNOBLAUCH	ARMIN R	DR	WAHNBACHTALSPERRENVE	KRONPRINZENSTRASSE 1	SIEGBURG	5200	GERMANY
KUHON	MICHEL		CARGILL		ST-GERMAIN-EN		FRANCE
KOUTCHOUK	MICHEL	APPLICATION & DEVELP	AIR FRANCE	DOMAINE DE VILGENIS	MASSY	91000	FRANCE
KRYNEN	ANTON P.J.	SYSTEM-MANAGER	U.N.U.VERKOOPGROEP	DE BOELELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
KUTNER	BENJAMIN	VICE PRESIDENT	AUTOMATED BUSINESS S	1649 W.BROAD STREET	RICHMOND VA	23220	USA
KWAVNICK	MYER	MANAGER D.P.	MONTREAL CHILDREN HO	2300 TUPPER	MONTREAL	QUH9B 2B6	CANADA
LAFFERT	NEB		HEWLETT PACKARD	CH. CHATEAU BLOC 19	LE LIGNON/GE	1219	SWITZERLAND
LAGERLOEV	ANNE BRIT	MANAGER SYS	NORGES ELEKT MATKONT	GAUSTADALLEN 30	OSLO 3		NORWAY
LAIDIG	KLAUS D.		HEWLETT PACKARD		BOEBLINGEN		GERMANY
LAIHO	MARTTI	SYSTEM MANAGER	PORASTO OY AB	TOOLONTULLINKATU 8	HELSINK	SF-00250	FINLAND
LANDUCCI	PAUL	SYSTEM ENGINEER	SERIC	120,RUE DE JAVEL	PARIS	75015	FRANCE
LASSEN	CARL CHRIST	DIRECTOR M.SC.	DOMI A/S	SDR.RINGVEJ 35	GLOSTRUP	DK 2600	DENMARK
LAUR	R.	JOURNALISTE					
LEE	MAX G.	OPNS SUPPORT MGR	HEWLETT PACKARD	5 AV RAYMOND CHANAS	EYBENS	ISE38320	FRANCE
LEFEBVRE	BERNARD	INFORMATICIEN	BONNETERIE CEVENOLE	1001 RUE REPUBLIQUE	GRANGES -LES-	07500	FRANCE
LEVANDOWSKI	STEPHEN A.	ASST. SYSTEMS MANAGE	LEVI STRAUSS EUROPE	AVENUE LOUISE 427	BRUXELLES	1060	BELGIUM
LEWIN	BOB	EUROPEAN SEO MANAGER	HEWLETT PACKARD	7 RUE DU BOIS DU LAN	MEYRIN GENEVA	1217	SWITZERLAND
LIEVENS-SOUDA	CLAUDE		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
LINDROOS	JARKI		AHLSTROM	ETELAESPLANADI 14	HELSINKI 13		FINLAND
LINDROOS	MAGNUS	EDP-PLANNER	JAAKKO POYRY GROUP	KAUPINTIE 3	HELSINKI 40	SF-00400	FINLAND
LOIZELEUR	ALAIN	ANALYSTE-PROGRAMMEUR	C.A.M.I.F.	TREVINS DE CHAURAY	NIORT CEDEX	79039	FRANCE
LORREN	LARRY C	U K SE MANAGER	HEWLETT PACKARD	KING STREET LANE	WINNERSH BERK	RG11 5AR	U.K.
LOSCH	LYDIA	INFO SYST ANALYST	BOSE CORP.	100 MOUNTAIN RD.	FRAMINGHAM (M		USA
LOVEGROVE	FRANK		CLAYBROOK COMPUTING	72 BROOK STREET	LONDON W1		U.K.
LUMB	ARTHUR C	SPECIALIST	PROCTER &GAMBLE CO	7162 READING ROAD	CINCINNATI OH	45222	USA
LUND	PER GO	CHIEF RES.ENGINEER	SHIP RES.INST.OF NOR	HAAKON HAAKONSENSGT.	TRONDHEIM	7000	NORWAY
MAJOR	FREDERIK	RESEARCH ENGINEER	SHIP RES.INST.OF NOR	HAAKON HAAKONSONSGT.	TRONDHEIM	7000	NORWAY
MAMPAEY	HARRY HAJ	TECHNICAL ENGINEER	CONT.AIR CONDITIONIN	HOF TER BECKENDREEF	WILRIJK	ANTB-2610	BELGIUM
MANEWAL	SANDRA		LIBERTY COMMUNICATIO	2225 COBURG ROAD	EUGENE (OREGO		USA
MANWELL	GERALD E	COMPUTER SYSTEMS MGR	TURBO RESOURCES LTD	1019 7 AVE SW	CALGARY ALBER	T2P 1A8	CANADA
MARAULT	GERARD	ORGANISATEUR CONSEIL	CSBP	131 AVENUE DE WAGRAM	PARIS	75017	FRANCE
MARCONTES	SERGIO		PROMON ENGENHARIA SA	AVENIDA NOVE DE JULH	SAO PAOLO	4939 0140	BRASIL
MARIOTTI	FRANCO	GL MANAGER HP EUROPE	HEWLETT PACKARD S.A	7 RUE DU BOIS DU LAN	MEYRIN GENEVA	1217	SWITZERLAND
MARQUIZEAUD	JACQUES	PUBLIC RELATIONS MGR	HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
MARTENS	DITLEV	CAND.DIPL.	SHIP RES.INST.OF NOR	HAAKON HAAKONSENSGT.	TRONDHEIM	7000	NORWAY
MASTRIPIERI	SERGIO	SENIOR SYSTEM ENGINE	HEWLETT PACKARD	VIA G.DI VITTORIO 9	CERNUSCO S/N	20063	ITALY
MATHEWS	STEVEN		BLACK & DECKER	MAIDENHEAD	BERKS		U.K.
MC AFEE	BILL	PRES.	FUTURA SYSTEMS INC	BOX 3485	AUSTIN TEXAS	78764	USA
MC CRACKEN	ED	GENERAL MANAGER GSD	HEWLETT PACKARD	19950 PRUNERIDGE	CUPERTINO CA	94560	USA
MELENNEC	J.L		ELF FRANCE	CENTER DE RECHERCCHE	CENTRE RECH,B	69360	FRANCE
MELVIN	PETER		CMC ASSOCIATES	19950 PRUNERIDGE	CUPERTINO CA	94560	USA
MEMMI	FERNAND						
MEZELLE	YANNIK		HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
MILLER	GORDON W.	SERVICES MANAGER	LONDON BUS. SCHOOL	SUSSEX PLACE	LONDON	NW1 4SA	U.K.
MINNS	ROBERT		QUASAR SYSTEMS LMTD		OTTAWA	ONTARIO	CANADA
MOELLER	CHRIS P	MEMBER TECH STAFF	HEWLETT PACKARD GSD	19447 PRUNERIDGE AVE	CUPERTINO CA	95014	USA
MOEN	JAN	SENIOR PROGRAMMER	RAET	GRONINGENSINGEL 1	ARNHEM		NEDERLAND

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
MONTLAHUC	GUY		HEWLETT PACKARD		ORSAY		FRANCE
MORY	BERTRAND	EDP MANAGER	RHODAFIN S.A	6 R.ANATOLE DE FORGE	PARIS	75017	FRANCE
MULAS	MR		CENTRO ELABORAT.DATI	COSTA ESMERALDA	PORTO CERVO	07020	ITALY
MULHANEY	DANIEL W.	STAFF MANAGER	HEWLETT PACKARD CO.	1203 114TH S.E.	BELLEVUE,WASH	98004	USA
MUSCARA	ANTONIO		AGUSTA	COSTA DI SAMARATE	CASCINA	21017	ITALY
MYLLE	CHRISTIAN		STE HAECON NV	MARATHONSTRAAT 12	GENT	B-9000	BELGIUM
NEDERHOF	VIC	CONSULTANT MANAGER	UI MANAG. CONSULT BV	POST BOX 760	ROTTERDAM	3000	NEDERLAND
NEIDHARDT	RUDOLF G	HEAD ORG/EDP DEPT	UNILEVER SCHWEITZ AG	FOERRLIBUCKSTR.10	ZUERICH	CH 8031	SWITZERLAND
NERDAL	JAN	PRODUCT MANAGER	NARDATA	GYMNASIEVAGEN 10	SKELLEFTEA	93101	SWEDEN
NIEDERHAUSER	BERNARD		HEWLETT PACKARD SA	7,RUE DU BOIS-DE-LAN	MEYRIN/GENEVE		SWITZERLAND
NOLAN	LAWRENCE	SUPERVISOR,SCIENTIFI	MCDONNELL DOUGLAS AU	P. O. BOX 516	ST LOUIS MIS	63166	USA
NONNENBERG	ALAN		HEWLETT PACKARD				GERMANY
NORDBERG	RAGNAR L E	DOC.	KLIN. KEM. C-LAB.	SAHLGREN'S HOSPITAL	GOTHENBURG	S-41345	SWEDEN
O'BRIAN	ROBERT	SECRETARY HPGSUG	HEWLETT PACKARD				AUSTRALIA
OBRIST	LINDA OL	ANALYSTE PROGRAMMATE	STE GENERALE DES MIN	31 RUE DU MARAIS	BRUSSELS	1000	BELGIUM
OLAUSSEN	DAVID EVEN		STATE TECH. COLLEGE		KONGSBERG ING		NORWAY
OLIVERIO	JAMES G	SYSTEMS SUPERVISOR	GTE DATA SERVICES	P.O. BOX 1548 DCNI07	TAMPA FLORIDA	33601	USA
OLSSON	PER OLOF PE	SYSTEM MANAGER	NARDATA	GYMNASIEVAGEN 10	SKELLEFTEA	93101	SWEDEN
OPPLIGER	MR		B.C.V	PLACE ST FRANCOIS	LAUSANNE		SWITZERLAND
OSCARSSON	HANS O.	DATA PROCESSING MGR	AB TRAV OCH GALOPP	BOX 1733	STOKHOLM	S-11187	SWEDEN
OUIN	GEORGES		HEWLETT PACKARD				FRANCE
PALEY	MAURICE	MGR.COMPUTER DEPT.	BOWATER CORPORATION	BOWATER HOUSE	KNIGHTSBRIDGE	SW1	U.K.
PAQUET	PIERRE		BG CHECO	110 CREMAZIE	MONTREAL QUEB		CANADA
PAULIN	CL.MARIE		ESC LYON	23 RTE DE DARDILLY	ECULLY	69130	FRANCE
PEFFERS	KEN		LEVIS-STRAUSS				BELGIUM
PERRIER	CHRISTINE		ESC LYON	23 ROUTE DE DARDILLY	ECULLY	69130	FRANCE
PIAZZA	J.C.		ALLIANCE INFORMATIQU	TOUR NEPTUNE CEDEX 2	PARIS LA DEFE	92086	FRANCE
PINEZ	EMANUEL	DIRECTOR	IEI SA	24 CH SARASIN	6D SACLONEX G		SWITZERLAND
POPELIN	ALAIN	TECHNICAL MGR	SERIC	120,RUE DE JAVEL	PARIS	75015	FRANCE
PORAT	ELIK	SYSTEM ENGINEER	HEWLETT PACKARD	19950 PRUNERIDGE	CUPERTINO CA	94560	USA
POT	DANIEL		HEWLETT PACKARD		GRENOBLE		FRANCE
POULSEN	POUL A.	ING.	ELKRAFT,P-EDB	PARALLELVEJ 19	LYNGBY	DK 2800	DENMARK
PRELLE	JEAN		ESC LYON	23 ROUTE DE DARDILLY	ECULLY	69130	FRANCE
RANDS	RICHARD D	MIS MANAGER	HEWLETT-PACKARD LTD	KING STREET LANE	WINNERSH BER	RG11 5AR	U.K.
RAVIT	J.Y.	RESP.DEPT.INFORMATIQ	ESC LYON	23 ROUTE DE DARDILLY	ECULLY	69130	FRANCE
REGO	ALFREDO	PROFESSOR	REGO SOFTWARE PTY	CALLE DEL ARCO 24	ANTIGUA		GUATEMALA
RICE	JOHN M	SYSTEMS DEV. MANAGER	CROWN DEC. PRODUCTS	CROWN HOUSE HOLLINS	DARWEN LA	BB3 0B9	U.K.
RIEDL	NORBERT F	DATA BASE ADMINISTRA	R.M.I.T.	124 LA TROBE STREET	MELBOURNE VI	3000	AUSTRALIA
RINGEN	DONALD L		HEWLETT PACKARD	KING STREET LANE	WINNERSH BERK	RG11 5AR	U.K.
ROBINSON	JOEL H	MGR INFORMATION SYS	GEORGE WIMPEY CAN LT	80 N QUEEN STREET	TORONTO ONT	M8Z 2C9	CANADA
ROGERS	BARRY H	SYSTEMS ANALYST	ILEC	120 S RIVERSIDE PLAZ	CHICAGO ILL	60606	USA
RUSSELL	JIM D	PRESIDENT	COMPUTER SERVICES CO	75 MANHATTAN DR#107	BOULDER CO	80303	USA
SADLER	K.S		MMDS INTERNATIONAL	91-93 FARRINGTON ROA	LONDRE	EC 1M 3LB	U.K.
SAGER	KURT E	HEAD OF DP	CIBA-GEIGY PHOTOCHEM	INDUSTRIESTRASSE 15	F R I B O U R	CH-1701	SWITZERLAND
SAILER	MANFRED	PROD. MANAGER	HEWLETT PACKARD GMBH	HERRENBERGERSTR. 110	BOEBLINGEN	7030	GERMANY
SAINT-GIRONS	ARNAUD	INFORMATICIAN	NESTLE	CASE POSTALE 88	LA TOUR DE PE	1814	SWITZERLAND
SAVATOWSKY	MR	SYSTEM ENGINEER	HEWLETT PACKARD	ZI COURTABOEUF	ORSAY	91401	FRANCE
SCHEHNER	MR		HEWLETT PACKARD		WIEN		AUSTRIA
SCHICK	JOHN A	SENIOR PROGRAMMER	WM.C.BROWN CO	2460 KERPER BLVD	DUBUQUE IA	52001	USA
SCHILD	MARIUS		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
SCHOENTHALER	OTTO M.	SYSTEMS ENGINEER	HEWLETT PACKARD	V.HEUVEN GOEDHARTLAA	AMSTELVEEN		NEDERLAND
SCHWAGER	A.		HEWLETT PACKARD				USA
SCHWEICKHART	REYNOLD		UNIVERSITY ST CLARA		SANTA CLARA	CALIFORNI	USA
SENANT	PIERRE		COGELOG	1 RUE DES QUINCONCES	GIF SUR YVETT	91190	FRANCE

NAME		TITLE	COMPANY	ADDRESS	CITY	ZIP	COUNTRY
SEVEBORG	KARL		AB BOFORS	POST BOX 900	B O F O R S	690 20	SWEDEN
SHOUP	BRYAN D	DATA PROCESSING MANA	SII DYNA DRILL	1771 DEERE	IRVINE CALI	92713	USA
SLAGER	HENK	SYSTEM PROGRAMMER	AGO VERZEKERINGEN	NIEUWESTAD 9	LEEUWARDEN	8911 CG	NEDERLAND
SMITHERS	JEREMY	SENIOR RESEARCH OFFI	LONDON BUSINESS SCHO	SUSSEX PLACE	LONDON	NW1	U.K.
SPALDING	KENNETH G	PROJECT MANAGER	HEWLETT PACKARD	19950 PRUNERIDGE	CUPERTINO CA	94560	USA
SPIERO	JAN G	PROGRAMMER	MAKRO INTERNATIONAAL	CHURCHILLLAAN 11	UTRECHT	3527 GV	NEDERLAND
SPORKEN	HEIN		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
STAMBAUGH	JAN		BOARD OF HPGSUG				USA
STANTINA	MARCEL		HEWLETT PACKARD	ZI CONTABOEUF	ORSAY	91401	FRANCE
STAPLETON	WALTER L	DIRECTOR IS&S*	MACMILLAN BLOEDEL LT	1075 WEST GEORGIA ST	VANCOUVER B.C	V6E 3R9	CANADA
STARK	JOHN A	SYSTEMS ENGINEER	HEWLETT PACKARD	1203 114TH S.E.	BELLEVUE,WASH	98004	USA
STEFFENS	JEAN	SYSTEM-MANAGER	U.N.U. VERKOOPGROEP	DE BOELELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
STENSETH	PER OLAV	SYSTEMS ENGINEER	HEWLETT PACKARD NORG	OSTERNOALEN 18	OSTERAS	1345	NORWAY
STIEFEL	EDUARDS J.	HEAD OF ORG/EDP	HABEGGER AG	INDUSTRIESTR.2	T H U N	CH-3601	SWITZERLAND
STUART	PETER		HEWLETT PACKARD		GRENOBLE	38000	FRANCE
SULLIVAN	JOHN J	DATA PROCESSING MGR	SIPPICAN CORP	1033 MASS AVE	CAMBRIDGE MA	02138	USA
SUNI	MARKKU		HELSINKI SCHOOL OF E	RUNEBERGIN KAT.14,16	HELSINKI	00100	1FINLAND
SZEPANSKY	ROBERT	MGR SYSTEMS DEVELOPM	AGO VERZEKERINGEN	NIEUWESTAD 9	LEEUWARDEN	8911 CG	NEDERLAND
TABOADA	NAPOLEON	GENERAL MANAGER	PRODASA LA CONSTANCI	P.INDEPENDENCIA Y 20	SAN SALVADOR		SALVADOR
TAGEL	TOIVO	DIRECTOR	DAFA	FACK	STOCKHOLM	S-10026	SWEDEN
TAHTINEN	MATTI	MR.	ENSO-GUTZEIT OY	ADP DEPARTMENT	IMATRA 80	SF-55800	FINLAND
TASS	JOHN R	COMPUTER SERVICES MG	WARWICK PUMP & ENG.	OXFORD ROAD	BERINSFIELD O	OX9 8LZ	U.K.
TAYLOR	JAMES	CONSULTANT		6 WEST OVER ROAD	LONDON	SW18	U.K.
TETI	FRANK A	A.PROGRAMM.OFFICER	MORGAN GUARANTY TRUS	23 WALL STREET	NEW YORK N.		USA
THOMAS	J.DIDIER		HEWLETT PACKARD		ORSAY		FRANCE
THOMPSON	DAVE A	DATA CENTER OPR MANA	HINDERLITER ENERGY E	1240 N HARVARD	TULSA OKLAHOM	74110	USA
TOBACK	BRUCE		A.L.S. CORPORATION	1400 N BAXTER STREET	ANAHEIM	CA 92806	USA
TREGUER	HERVE	SYSTEM ENGINEER	HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
TURNER	EDWARD	PRODUCT SUPPORT MGR.	HEWLETT PACKARD	19447 PRUNERIDGE AV	CUPERTINO CA	95014	USA
USSUNET	GUY C.	FOR OP&FIN MGR	SMITH INT'L FRANCE	RUE A.SIGONNEAU	LE BLANC MESN	93153	FRANCE
UTZ	WALTER		HEWLETT PACKARD GMBH	HERRENBERGER STRASSE	BOEBLINGEN	7030	GERMANY
UUSITUPA	SEPPO M I	EDP-MANAGER	GEOL.SURVEY OF FINLA	KIVIMIEHENTIE 1	E S P O O 15	SF-02150	FINLAND
VAN AUSDALL	CHUCK		BOARD OF HPGSUG				USA
VAN DE PEPP	F.		MENDES GANS NV	HERENGRACHT 619	AMSTERDAM	1017 CE	NEDERLAND
VAN DER ENDE	HYLKE	SYSTEM PROGRAMMER	AGO VERZEKERINGEN	NIEUWESTAD 9	LEUUWARDEN	8911 CG	NEDERLAND
VAN IEPEREN	KEES	SYSTEMS MANAGER	MAKRO INTERNATIONAAL	CHURCHILL-LAAN 11	UTRECHT UTREC	3527 GV	NEDERLAND
VAN RAMDERAAT	BOB		HEWLETT PACKARD		AMSTERDAM		NEDERLAND
VAN ROOYEN	MARCEL		NOVA AUTOMATION C.	NEDEREIND 30	NIEUWEGEIN		NEDERLAND
VAN WINSEN	THIJS P.H.	PROGRAMMEUR	U.N.U. VERKOOPGROEP	DE BOELELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
VASCHALDE	ALAIN		HEWLETT PACKARD	CHEMIN DES MOUILLES	ECULLY	69130	FRANCE
VASCONCELLOS	MR		WHERE ENDS MEET	5926 WHITNEY STREET	OAKLAND CA	94609	USA
VESTER	SJEF G.	PROGRAMMER ANALYST	AGO VERZEKERINGEN	NIEUWESTAD 9	LEEUWARDEN	8911 CG	NEDERLAND
VEZINA	GILLES E	SYSTEMS ANALYST	DOMTAR CHEMICALS	395 MAISONNEUVE BLVD	MONTREAL	QUH3A 1L6	CANADA
VIDELAINE	THIERRY		HEWLETT PACKARD		ORSAY		FRANCE
VITRON	PHILIPPE	O.R.MANAGER	CREDIT LYONNAIS	10 RUE ST AUGUSTIN	P A R I S	75002	FRANCE
VLUG	HAGE	PROGRAMMER	HEWLETT PACKARD BV	VAN HEUVEN GOEDHARTL	AMSTELVEEN	1101 KK	NEDERLAND
VROEGOP	PETER	EDP MANAGER	VROEGOP-RUHE EN CO B	CENTRALE MARKT 51	AMSTERDAM	1051 KL	NEDERLAND
WADEHN	GOORG	JOURNALISTE		SARRELWEG 33,3004 IS	HANNOVER		GERMANY
WALBYE	HANS	MANAGING ENGENIER	NORGES ELEKT MATKONT	GAUSTADALLEN 30	O S L O 3		NORWAY
WALDER	ESTEBAN	CIA ASS DATA CENTER	PROCESASEG SA				VENEZUELA
WEENINK	ANTON P.M.	SR.SYSTEEMONTWERPER	U.N.U.VERKOOPGROEP	DE BOLELAAN 7	AMSTERDAM	1083 HJ	NEDERLAND
WILLIAMS	ALAN F	SALES MANAGER	HEWLETT PACKARD	PRIVATE BAG	WENDYWOOD SA	N2144	SOUTH AFRICA
WING	MR	COMPUTER MANAGER	BRIT.NATIONAL OIL CO	150 ST.VINCENT STREE	GLASGOW	G2 5LJ	U.K.
WINTER	WILLIAM C.	SENIOR SECTION MANAG	MCDONNELL DOUGLAS AU	P. O. BOX 516	ST LOUIS	M63166	USA

```
   NAME                          TITLE                COMPANY                 ADDRESS                CITY         ZIP     COUNTRY

WISEMAN      DAVE B        TIMESHARING MANAGER   C U ASSURANCE           431 GODSTONE ROAD      WHYTELEAFE SUCR3 0YQ  U.K.
WOLF         PETER V.D.    SYSTEM-MANAGER        NESTLE NEDERLAND B.V    HILVERSUMSTRAAT 316    AMSTERDAM     1024 LG  NEDERLAND
WOLFF        FRITZ                               ANMERCO SA COMPUTER                                                   SOUTH AFRICA
WRIGHT       GORDON                              DE ZOETE & BEVAN        25 FINSBURY CIRCUS     LONDON EC2             U.K.
WULF         ROBERT                              INTL. LABOUR OFFICE                            GENEVA                 SWITZERLAND
YANSOUNI     CYRIL                               HEWLETT PACKARD                                GRENOBLE               FRANCE
ZALISK       RICHARD                             HEWLETT PACKARD         19950 PRUNERIDGE       CUPERTINO CA 94560     USA
```

CONFERENCE EXHIBITHORS

```
REPORT LISTE10


  HPGSUG 8TH INTERNATIONAL MEETING            EXHIBITOR'S LIST                            10/30/79     PAGE      1

     NAME                    TITLE                 COMPANY                  ADDRESS              CITY         ZIP        COUNTRY

 CORDELLE     YANN                              COGELOG                1 RUE DES QUINCONCES GIF SUR YVETT91190      FRANCE
 DAMM         JACK A.    PRINCIPAL              THE PALO ALTO GROUP    290 LUCERNE DRIVE    SUNNYVALE  CA94086      USA
 GREEN        ROBERT M   PRESIDENT              ROBELLE CONSULTING     #130-5421 10TH AVENU DELTA BC     V4M 3J9    CANADA
 HODSON       N.B                               CIE JOHN HOSKYNS       ASHTON LANE          SALE CHESHIREM33 1NP    U.K.
 KHURSH       AHMED      SYSTEMS MANAGER        MCMASTER UNIVERSITY    1200 MAIN ST W RM2D5 HAMILTON ONT L8S 4J9    CANADA
 MAJOR        FREDERIK   RESEARCH ENGINEER      SHIP RES.INST.OF NOR   HAAKON HAAKONSONSGT. TRONDHEIM    7000       NORWAY
 MELVIN       PETER                             CMC ASSOCIATES         19950 PRUNERIDGE     CUPERTINO CA 94560      USA
 NOLAN        LAWRENCE   SUPERVISOR,SCIENTIFI   MCDONNELL DOUGLAS AU   P. O. BOX 516        ST LOUIS  MIS63166      USA
 REGO         ALFREDO    PROFESSOR              REGO SOFTWARE PTY      CALLE DEL ARCO 24    ANTIGUA                 GUATEMALA
 VASCONCELLOS MR                                WHERE ENDS MEET        5926 WHITNEY STREET  OAKLAND CA   94609      USA
>
```